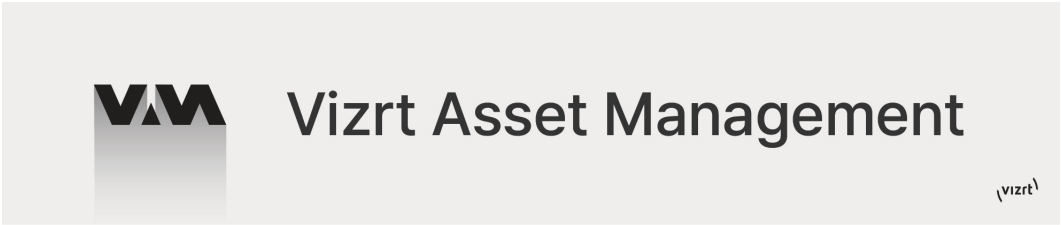
Vizrt Asset Management Administrator Guide

Version 1.0



Disclaimer

Vizrt provides this publication “as is” without warranty of any kind, either expressed or implied. This publication may contain technical inaccuracies or typographical errors. While every precaution has been taken in the preparation of this document to ensure that it contains accurate and up-to-date information, the publisher and author assume no responsibility for errors or omissions. Nor is any liability assumed for damages resulting from the use of the information contained in this document.

Vizrt’s policy is one of continual development, so the content of this document is periodically subject to be modified without notice. These changes will be incorporated in new editions of the publication. Vizrt may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time. Vizrt may have patents or pending patent applications covering subject matters in this document. The furnishing of this document does not give you any license to these patents.

Antivirus Considerations

Vizrt advises customers to use an AV solution that allows for custom exclusions and granular performance tuning to prevent unnecessary interference with our products. If interference is encountered:

- **Real-Time Scanning:** Keep it enabled, but exclude any performance-sensitive operations involving Vizrt-specific folders, files, and processes. For example:
  - C:\Program Files\[Product Name]
  - C:\ProgramData\[Product Name]
  - Any custom directory where [Product Name] stores data, and any specific process related to [Product Name].
- **Risk Acknowledgment:** Excluding certain folders/processes may improve performance, but also create an attack vector.
- **Scan Scheduling:** Run full system scans during off-peak hours.
- **False Positives:** If behavior-based detection flags a false positive, mark that executable as a trusted application.

Technical Support

For technical support and the latest news of upgrades, documentation, and related products, visit the Vizrt web site at www.vizrt.com.

Created on

2026/09/17

Contents

1 Legal

1.1 Legal

1.1.1 Disclaimer

Vizrt provides this publication “as is” without warranty of any kind, either expressed or implied. This publication may contain technical inaccuracies or typographical errors. While every precaution has been taken in the preparation of this document to ensure that it contains accurate and up-to-date information, the publisher and author assume no responsibility for errors or omissions. Nor is any liability assumed for damages resulting from the use of the information contained in this document.

Vizrt’s policy is one of continual development, so the content of this document is periodically subject to be modified without notice. These changes will be incorporated in new editions of the publication. Vizrt may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time. Vizrt may have patents or pending patent applications covering subject matters in this document. The furnishing of this document does not give you any license to these patents.

1.1.2 Disclaimer for AI-Assisted Features

This product enables end users (“You” or “Your”) who select this option to integrate their chosen external Artificial Intelligence (AI) tools, content and services (“Third Party AI”) through Vizrt’s MCP (Model Context Protocol) server (“Vizrt AI Assisted Features”). With respect to Your option to deploy your chosen Third-Party AI through Vizrt AI Assisted Features in any of Vizrt’s products or services (“Vizrt Products”), the following terms and conditions shall apply:

1.1.3 Third Party AI Models and Systems

Third Party AI—including but not limited to code, scripts, templates, configurations, and suggestions—is produced or made available by third-party developers and providers (“Third Parties”) of AI models and systems and/or your organization, and are not developed, owned, procured or controlled by or on behalf of Vizrt, its suppliers, agents, or Vizrt-authorized distributors or resellers of Vizrt Products. Your use of any Third Party AI may be subject to terms and conditions required by the applicable Third Party.

User Responsibility

With respect to your use of Third Party AI with Vizrt Products, including through Vizrt AI Assisted Features, you are solely responsible for the following undertakings, together with any other Third Party AI provider terms:

- Reviewing, understanding, and validating all Third Party AI before use.
- Testing the Third Party AI in non-production environments prior to deployment, to include:

- Backing up Third Party AI prior to integration with Vizrt Products;
- Ensuring Third Party AI (and your use thereof, with or without integration into Vizrt Products) complies with your organization's then-current policies and procedures related to AI, data-governance, IT, and security, and your organization’s coding standards;
- Verifying that Third Party AI is generated and used in compliance with the Third Party AI providers’/ deployers’ legal terms, policies and acceptable use guidelines, including those with respect to the infringement upon third-party intellectual property rights and other propriety rights; and
- Ensuring that any use of Third Party AI, including your use together with Vizrt Products, complies with applicable laws, regulations, your contractual obligations, including but not limited to, the EU AI Act and laws related to data privacy and artificial intelligence governance.

Any consequences resulting from the use of Third Party AI with Vizrt Products.

> ⚠ **Note:** In particular, we strongly recommend that you consider the risks associated with providing your inputs to a free or open source generative AI model or system where your inputs may be used to train the AI model.

No Endorsement or Warranty of Third Party AI

The availability of Vizrt AI Assisted Features does not constitute an endorsement, recommendation, or approval by Vizrt, its suppliers, agents, or Vizrt-authorized distributors or resellers of Vizrt Products, of any Third Party AI or the use of any specific AI provider, model, or service. Selection and use of Third Party AI is at your sole discretion and risk, and Vizrt disclaims any warranty with respect thereto.

1.1.4 Limitation of Liability

TO THE FULLEST EXTENT PERMITTED BY APPLICABLE LAW, NEITHER VIZRT NOR ANY OF ITS AFFILIATES, NOR THEIR RESPECTIVE OFFICERS, DIRECTORS, EMPLOYEES, AGENTS, SUBCONTRACTORS, OR ANY VIZRT-AUTHORIZED DISTRIBUTORS OR RESELLERS (COLLECTIVELY, THE “VIZRT PARTIES”), SHALL BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, CONSEQUENTIAL, SPECIAL, EXEMPLARY, OR PUNITIVE DAMAGES OF ANY KIND WHATSOEVER ARISING OUT OF OR RELATING TO THE USE, INTEGRATION, PERFORMANCE, OR NON-PERFORMANCE OF ANY THIRD-PARTY AI , SERVICES, OR COMPONENTS NOT PROVIDED BY VIZRT IN CONNECTION WITH ANY VIZRT PRODUCT OR SERVICE.

SUCH LIMITATION INCLUDES, WITHOUT LIMITATION, DAMAGES FOR: (A) SOFTWARE OR ALGORITHM DEFECTS; (B) DATA LOSS, CORRUPTION, OR EXPOSURE; (C) SECURITY OR PRIVACY BREACHES; (D) SYSTEM FAILURES OR PERFORMANCE DEGRADATION; (E) BUSINESS INTERRUPTION OR LOSS OF REVENUE, PROFITS, OR GOODWILL; OR (F) ANY OTHER DAMAGES, WHETHER CAUSED BY TORT (INCLUDING NEGLIGENCE), CONTRACT, STRICT LIABILITY, OR OTHERWISE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

AUTHORIZED VIZRT DISTRIBUTORS AND RESELLERS ARE EXPRESSLY INCLUDED IN THIS LIMITATION OF LIABILITY.

1.1.5 Antivirus Considerations

Vizrt advises customers to use an AV solution that allows for custom exclusions and granular performance tuning to prevent unnecessary interference with our products. If interference is encountered:

- **Real-Time Scanning:** Keep it enabled, but exclude any performance-sensitive operations involving Vizrt-specific folders, files, and processes. For example:
  - *C:\Program Files\[Product Name]*

    - *C:\ProgramData\[Product Name]*
    - Any custom directory where [Product Name] stores data, and any specific process related to [Product Name].
- **Risk Acknowledgment:** Excluding certain folders/processes may improve performance, but also create an attack vector.
- **Scan Scheduling:** Run full system scans during off-peak hours.
- **False Positives:** If behavior-based detection flags a false positive, mark that executable as a trusted application.

1.1.6 Technical Support

For technical support and the latest news of upgrades, documentation, and related products, visit the Vizrt web site at http://www.vizrt.com .

2 Introduction

This is the administrator documentation for **Vizrt Asset Management (VAM)**: installing it, operating it, and using the Configuration Service that ships with it.

Start with the section that matches the task in front of you. Each section stands on its own, so there is no need to read them in order.

2.1 Sections

2.1.1 Installer

End-to-end guide for installing VAM on a Windows host using the `vamctl` installer.

1. Installer Overview
2. Prerequisites
3. OIDC Authentication Provider Configuration
4. Running the Installer
5. Post-Install Validation
6. Upgrading
7. Uninstalling
8. Troubleshooting the Installer
9. Identity Provider Setup
10. VMware vSphere Deployment
11. Offline (Air-Gapped) Installation
12. Existing Kubernetes Cluster

2.1.2 Configuration Service

Operator guide for the **Configuration Service** — the in-cluster web UI used to manage packages, platform settings, certificates, backups and cluster health after installation.

1. Configuration Service Overview
2. Signing In
3. Managing Packages
4. Platform Settings
5. Backup and Restore
6. Cluster and Health
7. Troubleshooting the Configuration Service

2.2 Conventions

- Screenshots are taken from the latest released installer and configuration UI. If a screen in your installation looks different, check that you are on the matching VAM version (the installer's status bar and the configuration UI's footer both show the version number).
- Code-style boxes show literal text — file paths, command lines, or values you type into a UI field.

- Where a step requires elevated privileges or affects a running production cluster, it is marked with a **Warning** callout.

2.3 Getting Help

For support, contact your Vizrt support representative. When opening a support case, include:

- The VAM version (visible in the installer status bar and the Configuration Service footer).
- The installer log bundle, produced by **Help → Export logs** in the installer, or located at `%APPDATA%\Vizrt\vamctl\logs\` on the installer host.
- A description of what you were doing when the issue occurred.

3 Configuration Service

This section contains the following pages:

3.1 Configuration Service Overview

The **Configuration Service** is the web-based control panel for a running VAM installation. You use it for everything except installing VAM in the first place — managing packages, changing platform settings, running backups and checking cluster health.

3.1.1 When to Use the Configuration Service vs. the Installer

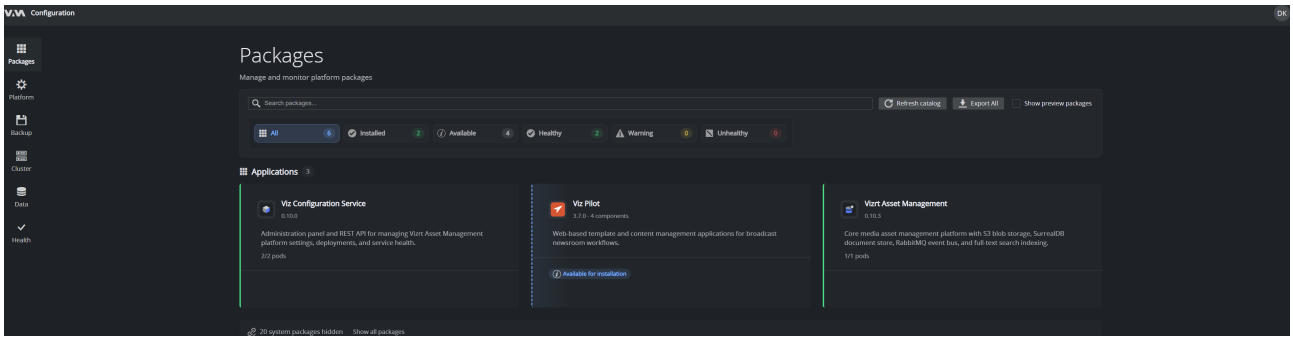
Task	Use this
First-time install of VAM	`vamctl` **installer**
Upgrade VAM to a new version	`vamctl` **installer**
Add or remove a cluster node	`vamctl` **installer**
Install / upgrade / uninstall a package	**Configuration Service**
Change platform settings (cert, OIDC)	**Configuration Service**
Take or restore a backup	**Configuration Service** (or installer for full restores)
Check pod / service health	**Configuration Service**
Manage licenses	**Configuration Service**

3.1.2 How to Open It

Browse to `https://<your-vam-host>/config` (the exact URL was shown on the installer's Finish screen). Sign in with the OIDC identity provider you configured during installation — see Signing in.

You do **not** need access to the installer host to use the Configuration Service; it runs inside the cluster and is reachable over the network like any other web app.

3.1.3 The Navigation Map



After sign-in, the left navigation has these sections:

Section	What it's for
Packages	Browse, install, configure, upgrade and uninstall VAM packages.
Platform	Authentication (OIDC), certificates (ADCS), networking, storage license, replication, advanced settings and the audit log.
Backup	Run and schedule backups; verify and restore them.
Cluster	Read-only view of the underlying Kubernetes cluster — pods, workloads, services, events.
Data	Browse the platform data store (in development).
Health	On-demand health scan of the whole installation.

Your account menu (the avatar in the top-right) is where you sign out.

The **Packages** view is the steady-state workspace — most day-to-day operation happens there.

> *The **storage license** (AiStor enterprise license) lives under **Platform → Storage license**, not a separate top-level section.*

3.1.4 Core Concept: Packages

Everything VAM installs is a **package**. The core asset-management services are packages; optional connectors and integrations are packages; sample data is a package. The Configuration Service is shaped around this:

- The **catalog** shows packages available to install.
- The **package detail page** shows a single installed package: its current version, its configuration, its lifecycle actions (upgrade, reconfigure, uninstall) and its health.

This means that to configure VAM, or to upgrade a feature, you find the relevant package and operate on it. There is no separate *settings* surface for individual features.

3.1.5 Where to Go Next

- First time signing in: Signing in.
- Day-to-day operation: Managing packages.
- After install: configure backups in Backup & restore.

3.2 Signing In

The Configuration Service uses your organization's identity provider (OIDC) for sign-in. Which provider depends on how VAM was installed:

- **Bundled Keycloak**: the installer deployed a Keycloak instance for you. Use the admin credentials shown on the installer's Finish screen for the first sign-in.
- **External OIDC**: Microsoft Entra ID (Azure AD), Okta, Auth0, or your own Keycloak. Use the same credentials you would for any other application connected to that provider.

3.2.1 First Sign-in

1. Open `https://<your-vam-host>/config` in a browser.
2. Click **Sign in**.
3. You are redirected to your identity provider's sign-in page.
4. Enter your credentials.
5. After successful sign-in you are redirected back to the Configuration Service and land on the **Packages** view.

Change the Bundled Keycloak Admin Password

Sign-in is handled by your identity provider, so passwords are changed **there**, not in the Configuration Service. If you used the bundled Keycloak, **change the initial admin password immediately** in the Keycloak account/admin console (its URL is shown on the installer's Finish screen).

3.2.2 Adding Users

How you add users depends on your identity provider.

Bundled Keycloak

1. Open the bundled Keycloak admin console (its URL is shown on the installer's Finish screen).
2. Sign in with the admin user.
3. Navigate to **Users → Add user**.
4. Fill in the user details, save and set a password under **Credentials**.
5. Assign roles under **Role mapping** — assign `vam-admin` to grant Configuration Service access (the Configuration Service requires this exact role).

The user can now sign in to the Configuration Service.

External OIDC

User management happens entirely in your existing identity provider — Azure AD, Okta, etc. The only VAM-specific step is to ensure users are assigned to the application (or to the right group) in your provider's console, **with the roles VAM expects**. For Microsoft Entra ID the `vam` role is mandatory — without it a user authenticates but cannot use VAM. See the Identity Provider Setup guide for the full role list and how to assign it.

3.2.3 Roles

The Configuration Service authorizes on a single role:

Role	Permissions
`vam-admin`	Full access to the Configuration Service: install / upgrade / uninstall packages, change platform settings, take and restore backups and view cluster health. The service requires this **exact** role on every request — a user without it can sign in but cannot use the Configuration Service (there is no read-only tier).

If your identity provider uses groups instead of roles, map those groups to the roles VAM expects **in your identity provider** (VAM reads the roles and groups from the OIDC token; there is no separate role-mapping screen in the Configuration Service).

3.2.4 Signing Out

Use the account menu (the avatar in the top-right) and choose **Sign out**. You are returned to the sign-in page. Depending on your identity provider's settings, your session at the provider itself may remain — sign out there too if you are on a shared computer.

3.2.5 Trouble Signing in

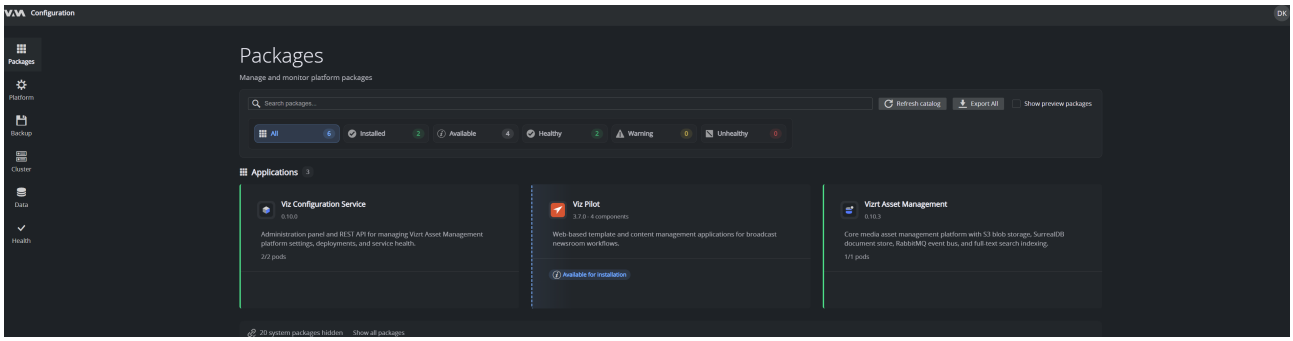
See Troubleshooting → Cannot sign in. For external-provider problems (redirect-URI mismatch, missing roles, expired secret, storage access denied), see Installer troubleshooting → Sign-in and authorization.

3.3 Managing Packages

Packages are the unit of installation in VAM. Everything you can add, configure, upgrade, or remove is a package.

3.3.1 The Packages View

The default landing page after sign-in lists every package known to the installation, grouped by category (Applications first). Each package is shown as a **card**.



Across the top you have:

- **Search**: filter the list by name.
- **Filter chips**: `All`, `Installed`, `Available`, `Healthy`, `Warning`, `Unhealthy` (each with a count) to narrow what's shown.
- **Refresh catalog**: re-pull the catalog from the registry so newly published packages / versions appear (no redeploy needed).
- **Export All** and **Show preview packages** for advanced use.

Each card shows the package's name, installed **version**, a short description, its **pod count** (for example, `2/2 pods`) and badges such as **Update available**. Infrastructure/system packages are hidden by default — use **Show all packages** to reveal them.

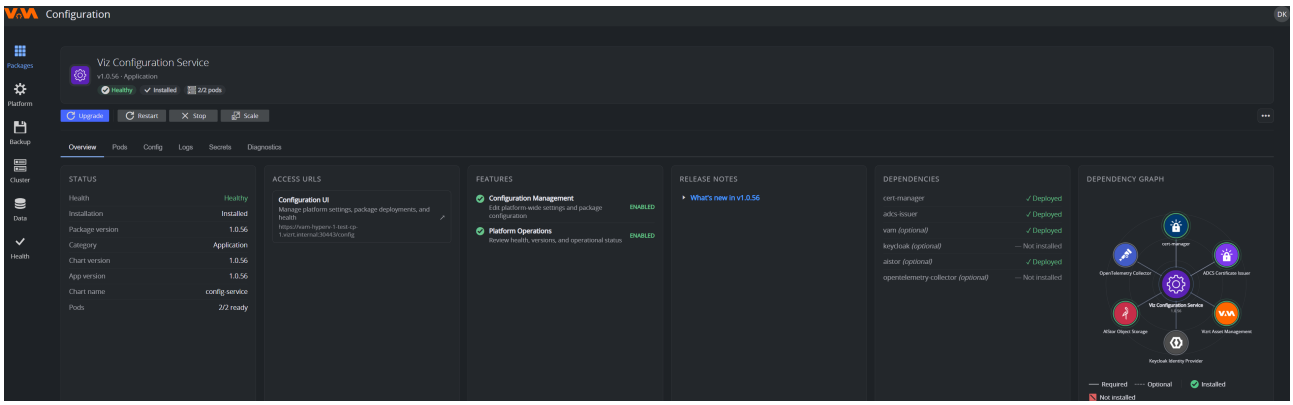
Click a card to open the **package detail view**.

3.3.2 Package Detail View

The package detail view is where you do everything for a single package. Lifecycle actions live in the button row at the top — **Upgrade**, **Restart**, **Stop**, **Scale** and an **…** overflow menu (Uninstall lives here when the package allows it). Below that is a set of tabs:

- **Overview**: status (health, installed version, chart/app version, pod readiness), access URLs, the package's enabled features, **release notes** (the "What's new" changelog), its dependencies and a visual **dependency graph**.
- **Pods**: the package's running pods and their status.

- **Config**: the package's settings, as a form generated from the package's published config schema (schema-validated).
- **Logs**: recent log output from the package's pods.
- **Secrets**: the secrets the package consumes (values are never shown).
- **Diagnostics**: additional detail for support/debugging.



3.3.3 Installing a New Package

1. On the **Packages** view, filter by **Available** (or use **Refresh catalog** first if you just published something).
2. Click the package card to open its detail view.
3. Read the description and check the **Dependencies** section / dependency graph. Any missing required dependencies are listed; the UI installs them in the right order if you accept.
4. Click **Install**.
5. Fill in the configuration form. Required fields are marked; the form does not let you submit until they are filled and valid. Some packages add gates here (for example a GPU check).
6. Click **Install** to start.
7. Live progress is shown step-by-step. On success, the package appears as **Installed** with status **Healthy**.

If the install fails, expand the failed step to read the diagnosis. Most failures are configuration problems — fix the values, then retry from the package detail page.

3.3.4 Configuring an Installed Package

1. Open the package's detail view.
2. Click the **Config** tab.
3. Edit values. As you change a field, the UI validates against the package's schema and shows inline errors if a value is out of range, the wrong type, etc.
4. Save:
    - **Save Configuration** persists the values without redeploying.
    - **Save & Apply** persists *and* rolls the change out to the running package.
    - **Reset to Saved** discards your unsaved edits.
5. Disruptive changes are applied in place — typically with no downtime.

Some configuration values (database passwords, OIDC client secrets) are sealed secrets; the UI shows `••••••••` for the current value and lets you replace it, never read it back.

3.3.5 Upgrading a Package

1. The package card / detail page shows an **Update available** badge when the catalog has a newer version. (Use **Refresh catalog** to pick up newly published versions.)
2. Click **Update**.
3. A dialog shows **what's changing — vX → vY**: the release notes / changelog for the new version and the configuration to review.
4. Click **Update** to start; live step-by-step progress is shown.

If an upgrade fails, it is **automatically rolled back** to the last known-good version and the dialog explains what happened — so a bad upgrade does not leave the package broken.

> ***Self-upgrade:*** *the Configuration Service can upgrade itself through this same flow. When it does, it briefly restarts and reconnects automatically and a dedicated progress dialog tracks the steps. Upgrading the* ***core VAM platform*** *(and adding/removing cluster nodes) is still done through the* `vamctl` *installer.*

3.3.6 Uninstalling a Package

1. Open the package's detail view.
2. Click **Uninstall**.
3. Read the warning — uninstalling deletes the package's data.
4. Type the package name to confirm.
5. Click **Uninstall**.

If other installed packages depend on this one, the UI refuses and lists the dependents. Uninstall them first (or, if intentional, override with **Force uninstall** — but only after reading the warning).

3.3.7 Reading Package Logs

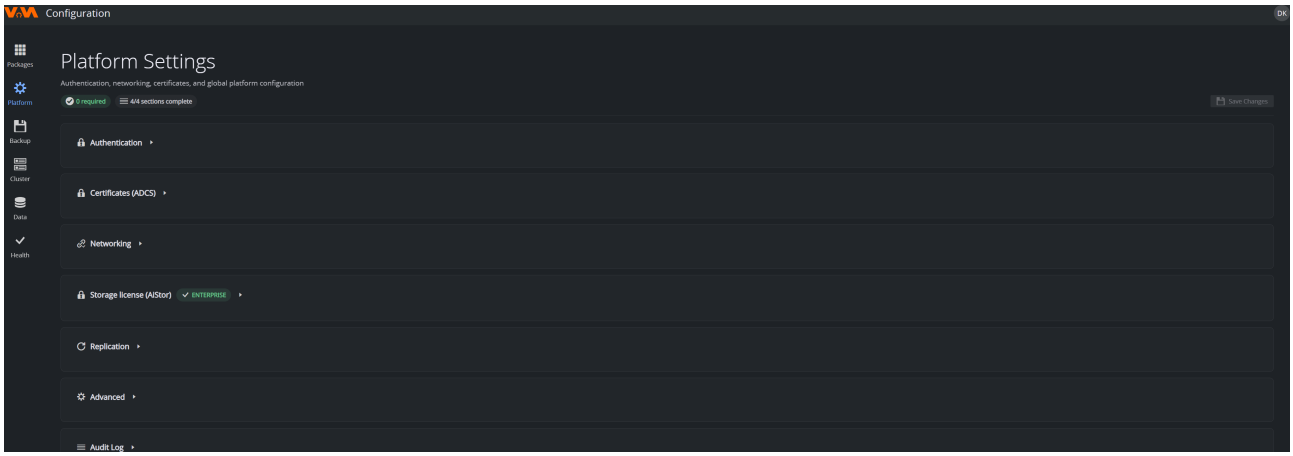
Open the **Logs** tab on the package detail view. The UI streams recent log lines from the package's pods. Filter by:

- **Severity**: show only errors / warnings / info / debug.
- **Pod**: if the package has multiple pods, pick one.
- **Time range**: last 5 minutes / 1 hour / 24 hours / custom.

For long-term log analysis (beyond what the UI buffer holds), contact your administrator about the cluster's observability stack.

3.4 Platform Settings

The **Platform** section controls settings that apply to the whole installation rather than to a single package: authentication, certificates, networking, the storage license, replication, advanced options and the audit log.



Platform settings are designed to be operable even when the VAM application itself is down — this is what makes recovery possible. You can sign in to the Configuration Service and fix platform settings even if VAM is reporting unhealthy.

Each area is a collapsible section. Open a section, edit its fields, then use **Save Changes** at the top-right.

3.4.1 Validate Before Apply, and the Safety Window

Platform changes are **validated before they are applied**, and risky changes have an **undo window** — this is what stops a typo from locking everyone out:

- **Pre-flight validation.** When you save, the change is checked against the live dependency first. A wrong or unreachable OIDC authority, or an unreachable ADCS endpoint, is **rejected up front** — nothing is written. A hostname that doesn't resolve in DNS yet is a **non-blocking warning** (the change still applies).
- **Pending window / auto-revert.** After an auth/cert change is applied, a banner asks you to confirm with **Keep new settings**. If you don't confirm within a few minutes (for example because the change locked you out), the change **automatically reverts** to the previous working values. You can also **Discard** immediately.

3.4.2 Authentication (OIDC)

Points every VAM service at your identity provider. Fields:

- **OIDC Provider URL**: the authority / issuer (for Microsoft Entra ID this is `https://login.microsoftonline.com/<tenant>/v2.0`).
- **Client ID** and **Realm**.
- **OIDC Client Secret**: write-only. A **Set / Not set** chip shows whether one is stored; leave the field empty to keep the current value, paste a new value to rotate it.

On save, the provider's discovery document is fetched and its `issuer` is checked against what you entered — a wrong host, realm, or tenant is rejected before any service is pointed at a broken authority.

3.4.3 Certificates (ADCS)

How VAM obtains TLS certificates from your Active Directory Certificate Services authority. Fields: **ADCS Server URL**, **Username**, **Password** (write-only, same Set/Not-set behavior as above), **Certificate Template** and **Issuer**.

A change to the URL, template, or credentials is **canary-validated**: the platform issues a throwaway certificate through a temporary copy of the issuer carrying the new values, and only if that succeeds does it update the live issuer and re-issue the affected certificates. A bad value leaves the running certificates untouched, and the banner tells you the validation failed.

- **Restart affected services now**: off by default. The new certificate is issued into its secret either way; this toggle only controls whether the consuming services are restarted immediately (to present the new certificate) or pick it up on their next restart.

3.4.4 Networking

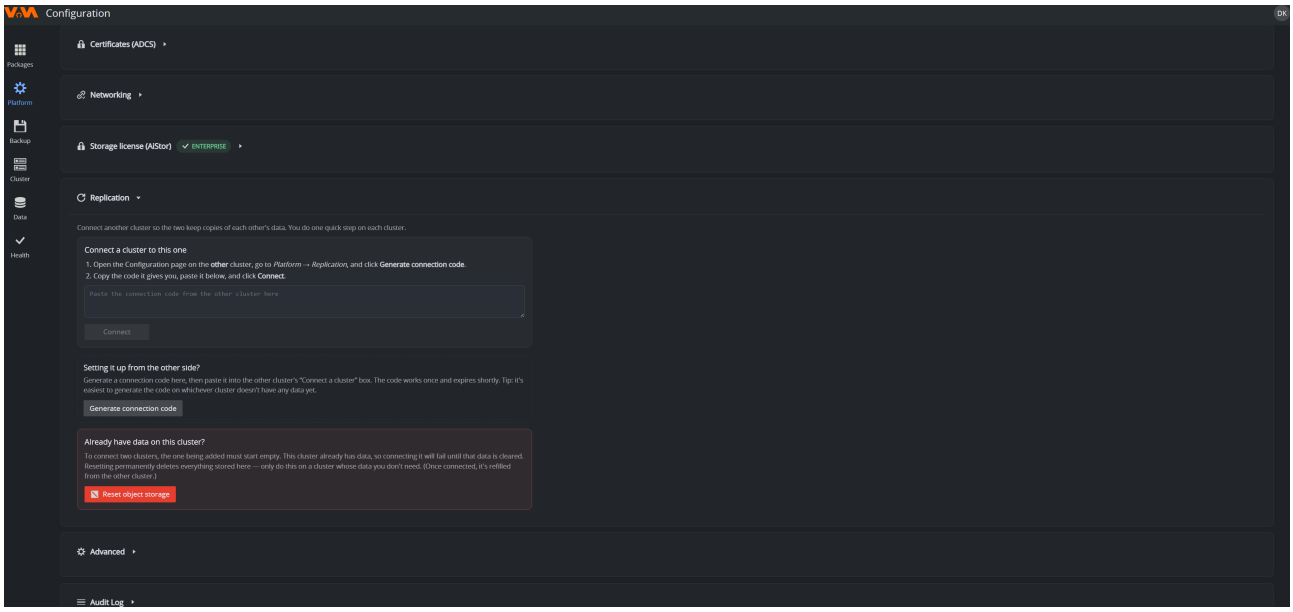
The external hostname and TLS settings clients use to reach the platform. On save, the hostname is checked in DNS — if it doesn't resolve yet you get a **non-blocking warning** (the change still applies) so you can add the DNS record afterwards.

3.4.5 Storage License (AiStor)

The enterprise object-storage (AiStor) license. The section header shows the current entitlement (for example, **ENTERPRISE**). Paste a new license here to install or replace it. Replication and other enterprise storage features are gated on a valid license.

3.4.6 Replication

Connect two clusters so they keep copies of each other's object data (active-active). You do one quick step on each cluster.



1. On the cluster that already has data, open **Platform → Replication** and click **Generate connection code**.
2. On the other cluster, paste that code into **Connect a cluster to this one** and click **Connect**.

The code is single-use and expires shortly. The cluster being *added* must start empty — the panel warns you and offers a reset if it already holds data. Once connected, the Replication view shows per-site status and an aggregate health summary and a **Disable** action tears the mesh down and removes the peer sites.

3.4.7 Advanced

Less-common global options (for example developer-mode exposure of internal services). Most installations never need to change these.

3.4.8 Audit Log

A record of platform-configuration changes — who changed what, and when. Use it to review recent administrative activity.

3.4.9 Trust Roots

Admin-uploaded root CA certificates that the platform trusts (for example an internal CA that signs the ADCS endpoint or other services VAM calls). Reached from **Platform → Trust Roots**.

1. Click **Upload PEM** and choose a PEM-encoded root certificate.
2. The root is added to the `vizam-trust-bundle` the platform distributes, and (best-effort) mirrored into the ADCS issuer's CA bundle.
3. **Restart workloads after change** (a checkbox, off by default) controls whether running workloads are restarted now to pick up the new trust, or on their next restart.

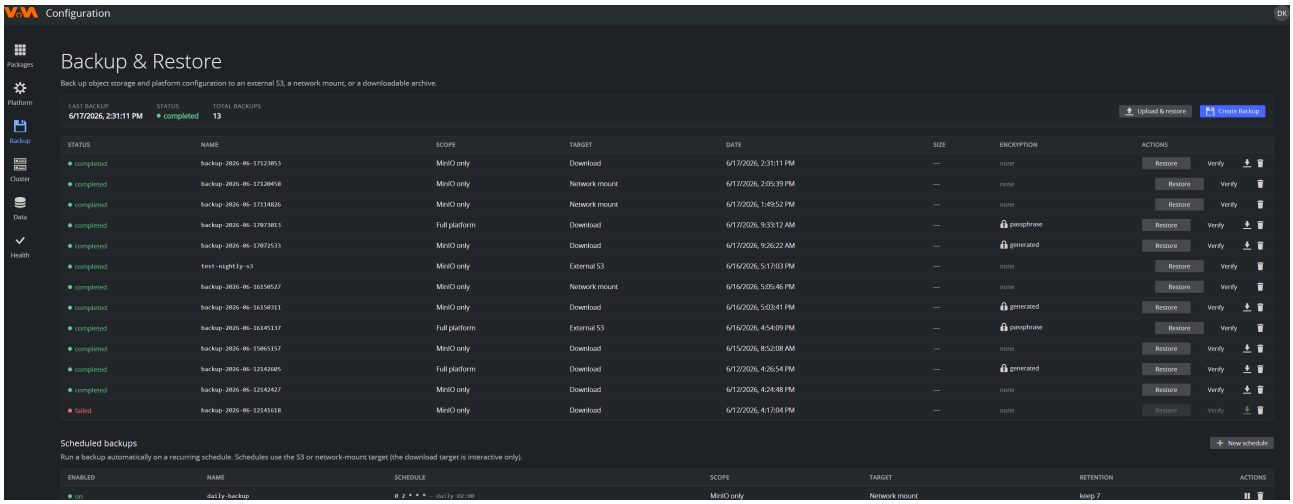
Each root is listed with its subject, expiry and SHA-256, with a **Delete** action to remove it.

3.4.10 If the Configuration Service Itself Won't Start

There is a `vamctl` command-line path for emergency platform-settings recovery without the web UI — contact support.

3.5 Backup and Restore

Backups capture the state of your VAM installation so it can be restored after data loss, accidental deletion, or a failed upgrade.



The **Backup** view lists every backup with its **status**, **scope**, **target**, date, size and **encryption**, plus a header summary (last backup, status, total count). Unlike earlier versions there is no separate *destination* to configure first — you choose the scope, target and encryption **each time** you create a backup.

3.5.1 What a Backup Contains — Choose the Scope

When you create a backup you pick a **scope**:

- **MinIO only**: the object-storage (AiStor/MinIO) data.
- **Full platform**: object storage **plus** platform configuration (package configuration, platform settings, trust roots) and, for the bundled Keycloak, its users/roles. External OIDC is owned by your provider and is not in scope.

3.5.2 Run a Backup

1. Open **Backup** and click **Create Backup**.
2. Choose the **scope** (MinIO only / Full platform).
3. Choose the **target**:
    - **Download**: produces a downloadable archive (interactive only).
    - **Network mount**: writes to an SMB/NFS share mounted into the cluster (see *Network-mount target* below).
    - **External S3**: an S3-compatible endpoint/bucket.
4. Choose **encryption**: **none**, a **generated** key, or your own **passphrase**. (Keep generated keys / passphrases safe — they are required to restore.)
5. Optionally add a label, then start. Live progress is shown; the new backup appears in the list when done.

Network-Mount Target

The Network-mount target needs a PersistentVolume that points at your SMB/NFS share. The Configuration Service can create the PV/PVC for you from the backup screen — provide the share path and credentials and it provisions the mount target (the SMB CSI driver must be installed for SMB). After that the share is selectable as a backup target.

3.5.3 Schedule Recurring Backups

In the **Scheduled backups** section, click **New schedule** and pick a frequency and retention. Schedules use the **S3** or **network-mount** target — the **Download** target is interactive-only and can't be scheduled.

3.5.4 Verify a Backup

Each backup has a **Verify** action that checks the backup's integrity **without doing a full restore**. Run it periodically, and especially before relying on a backup for a planned change.

3.5.5 Restore

> ***Warning**: restoring replaces the current installation's state. Changes made after the backup was taken are lost, and a restore involves downtime.*

From a Backup in the List

1. Find the backup and click **Restore**.
2. Review what the restore replaces, then confirm.
3. Wait for completion. The Configuration Service may briefly sign you out as services restart; sign back in afterwards.

From a Downloaded Archive

Use **Upload & restore** at the top of the page to upload a previously **downloaded** backup archive and restore it — useful when moving a backup between environments. You'll need the encryption key/passphrase if the archive was encrypted.

Full Disaster Recovery (New Cluster)

For restoring onto a freshly reinstalled cluster, use the `vamctl` installer's restore path rather than the Configuration Service — see Installer → Upgrade → Rollback.

3.5.6 Testing Backups

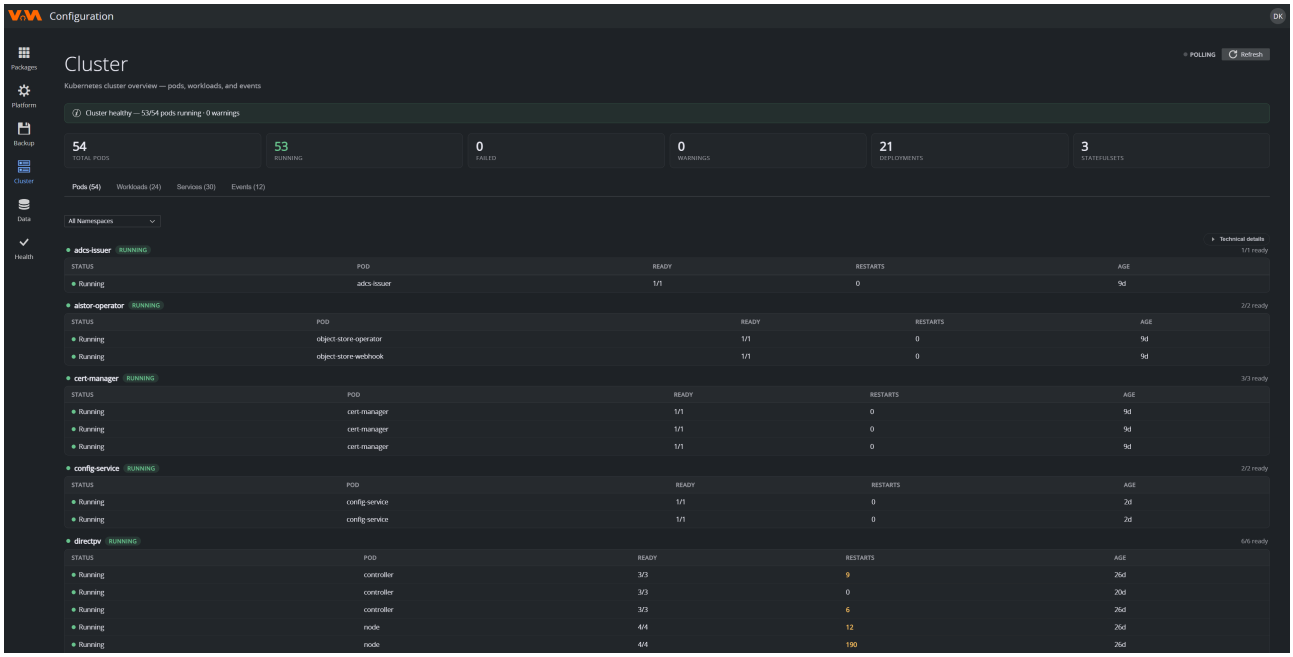
Untested backups are a liability. Periodically **Verify** your latest backups in-place, and for full confidence restore a recent backup onto a sandbox installation and confirm key data is present.

3.6 Cluster and Health

The **Cluster** and **Health** sections give operators and support a read-only window into the running installation. You don't need them for day-to-day VAM use.

3.6.1 Cluster

Cluster in the navigation is a live view of the Kubernetes cluster that hosts VAM. The header summarises **total pods**, **running**, **failed**, **warnings**, **deployments** and **statefulsets**, with an overall "Cluster healthy" banner. It refreshes automatically (live), and there's a **Refresh** button.



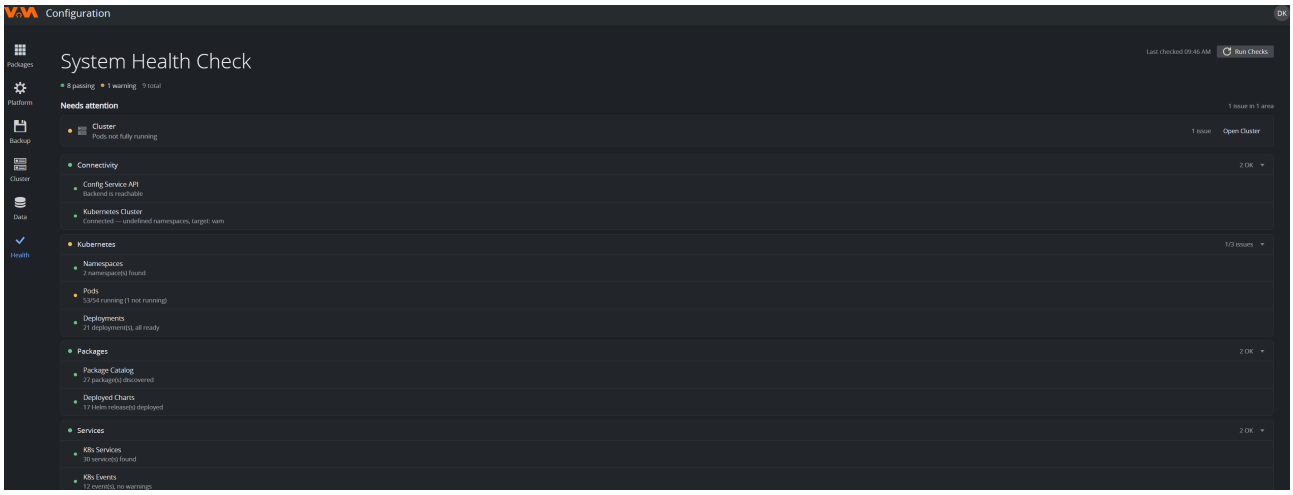
Tabs:

- **Pods**: every pod (filterable by namespace), grouped by workload, with ready count, **restart count** (non-zero is a warning sign) and age. A **Technical details** expander shows more.
- **Workloads**: deployments / statefulsets / daemonsets.
- **Services**: Kubernetes services.
- **Events**: recent cluster events.

Healthy looks like: all workloads green, zero pods in `CrashLoopBackOff` / `ImagePullBackOff` / `Failed` and only a few short-lived `Pending` pods during upgrades.

3.6.2 Health (System Health Check)

Health runs a coordinated health scan across the whole installation — the same kind of check the installer runs after deployment, available on demand. Click **Run Checks**.



Results are grouped into areas — **Connectivity**, **Kubernetes** (namespaces, pods, deployments), **Packages** and **Services** — each shown as OK or with a count of issues, plus a **Needs attention** summary at the top that links straight to the relevant area (for example, **Open Cluster**).

Run it after any upgrade or configuration change, when a user reports an issue you can't reproduce and before opening a support case.

3.6.3 When to Escalate

Open a support case if any of these are true:

- A node has been **Not Ready** for more than ~15 minutes and a restart doesn't fix it.
- A pod is in `CrashLoopBackOff` and its logs don't explain why.
- Health-check failures persist after applying the suggested remediations.
- The whole Configuration Service is unreachable.

3.7 Troubleshooting the Configuration Service

Common Configuration Service problems and how to resolve them. For installer-side issues see Installer → Troubleshooting.

3.7.1 Cannot Sign in

Symptom: clicking **Sign in** loops back to the sign-in screen, or the identity provider shows an error.

Resolve:

1. **Wrong credentials.** Try resetting your password through your identity provider, then retry.
2. **Account missing the required role.** The Configuration Service requires the `vam-admin` role. Check that your account is assigned `vam-admin` (in Keycloak or via group mapping in your external OIDC).
3. **OIDC misconfiguration.** If you recently changed the identity provider settings, recheck them in **Platform → Authentication**. The page validates the authority before applying, and an auth/cert change auto-reverts if you don't confirm it — so if you were just locked out, wait for the pending window to revert, or sign in from another device and **Discard**. In the worst case, contact support to reset OIDC via `vamctl`.
4. **Clock skew.** OIDC tokens are time-sensitive. Confirm the cluster nodes' time is synchronised (NTP).
5. **Redirect URI mismatch.** Confirm the redirect URI registered in your identity provider matches the Configuration Service's URL (`https://<your-vam-host>/config`) exactly — including `https` and no unexpected trailing slash.

3.7.2 Configuration Service Is Unreachable

Symptom: browsing to `https://<your-vam-host>/config` shows a connection error, certificate error, or 502 / 503.

Resolve:

1. **DNS / network.** Confirm the hostname resolves and the IP is reachable from your client.
2. **TLS certificate expired.** A browser certificate warning that wasn't there yesterday suggests an expired or replaced certificate. Use `vamctl` on the installer host to reissue.
3. **Service down.** Run **Health → Run Checks** (or, if the UI is unreachable, use `vamctl` on the installer host). If multiple services are down, the cluster itself may be unhealthy — see the **Cluster** view or the installer's cluster-health view.
4. **Ingress down.** If the cluster is healthy but the config service URL is unreachable, the ingress controller pod may have crashed. Contact support — restarting ingress is risky and support has tooling for it.

3.7.3 Package Install Fails

Symptom: installing a package reports an error part-way through.

Resolve:

1. Expand the failed step to read the diagnosis. Most failures are configuration-related (a required field with an invalid value, credentials that don't work).
2. Fix the configuration on the package's install form and retry.
3. If the error mentions **missing dependencies**, install the dependent packages first.

4. If the error mentions **registry / image pull**, the cluster cannot fetch the package's container images. Confirm the cluster has network access to the configured registry (registry settings are established at install time via `vamctl`).

3.7.4 Package Shows "Degraded" or "Failed"

Symptom: a package's status indicator is yellow or red on the Packages list.

Resolve:

1. Click the package to open its detail view.
2. Read the **Overview** status and the **Diagnostics** tab — they show the specific failing checks.
3. Read the **Logs** tab (and the **Pods** tab) for relevant error messages.
4. Common causes:
    - The package depends on another that is down.
    - The package's configuration references credentials that have rotated.
    - A backing store (database, object store) is unhealthy.

If you cannot diagnose from the package logs alone, run **Health → Run Checks** for a coordinated view.

3.7.5 "Configuration Save Failed: Schema Validation"

Symptom: editing a package's configuration and clicking save shows a validation error.

Resolve: the package's configuration schema rejected a value. The error message points to the specific field. Fix the value and retry. If the message is unclear, the package's documentation (visible on its detail view) explains valid values.

3.7.6 Backup to Network Mount / External S3 Fails

Symptom: a backup whose target is **Network mount** or **External S3** fails (the **Download** target is local and rarely fails).

Resolve:

- **Network mount**: confirm the SMB / NFS share is reachable from the **cluster nodes** (not just your workstation), the PV/PVC for the share exists (create it from the Backup screen if not) and the credentials have read+write. For SMB, the SMB CSI driver must be installed.
- **External S3**: confirm the endpoint URL is reachable from the cluster, the bucket exists and the access key has `s3:PutObject`, `s3:GetObject` and `s3:ListBucket` on the bucket.
- Use a backup's **Verify** action to confirm integrity of completed backups.

3.7.7 Storage License Shows Expired / Invalid

Symptom: **Platform → Storage license** reports the AiStor license is expired, invalid, or has the wrong feature set (enterprise features such as replication are gated on it).

Resolve:

1. Contact Vizrt to request a renewed or corrected license.
2. Open **Platform → Storage license**, paste the new license and save.

3. Confirm the entitlement (for example, **ENTERPRISE**) in the section header.

3.7.8 Getting Support

When opening a support case, include:

- The VAM version (Configuration Service footer).
- A current **Health → Run Checks** result.
- The installer log bundle from the installer host (if relevant).
- A description of what you were doing when the issue occurred and what you've already tried.

4 Installer

This section contains the following pages:

4.1 Installer Overview

Vizrt Asset Management (VAM) is a media asset management platform that runs as a set of containerised services on a Kubernetes cluster. You install it once on a Windows host using the `vamctl` installer; the installer brings up the cluster, deploys the VAM services and hands you a web UI for ongoing operation.

The installer makes substantial changes to the host and needs **Administrator** rights throughout. Before you run it on a machine you care about, understand that it:

- Enables and uses **Hyper-V**, and creates, reconfigures and **deletes** virtual machines and virtual switches. An install onto a host that already has VAM VMs re-provisions them, which destroys their contents.
- Runs a **rootful Podman machine on WSL 2** — used to build the Talos boot images and stage offline bundles, so it is required for every install — and issues `wsl --shutdown` while configuring it. That stops **every** WSL distribution on the host, so save any other WSL work first.
- **Formats disks** you assign to cluster storage. The Deployment Configuration step names each drive it will format before you confirm.
- Installs command-line tooling (kubectl, Helm, talosctl, Podman and others) onto the host.

Use a dedicated host for production. Read Prerequisites first.

4.1.1 What Gets Installed

A single VAM installation consists of:

Component	What it is
Kubernetes cluster	A small Kubernetes cluster (one or more nodes) that hosts VAM.
VAM core services	The asset-management services (ingest, search, storage backends).
Configuration Service	Web UI for managing the installation after setup is complete.
Supporting infrastructure	Object storage, document store, message broker, in-cluster registry and observability (metrics, logs, traces).
`vamctl` **on the install host**	The installer and lifecycle tool, kept on disk for upgrades.

You interact with the installation in three ways:

- `vamctl` **installer (Windows)**: used for first-time install, cluster lifecycle (add a node, upgrade Kubernetes), backup and upgrade to a new VAM version.
- **Configuration Service (web UI)**: day-to-day operation — installing optional packages, changing platform settings, taking backups, viewing cluster health.

- **End-user VAM applications**: the actual asset management apps your editors and journalists use; these are out of scope for this document.

4.1.2 Installation Modes

The installer offers three workflows from its start screen:

Workflow	When to use
Online installation	Install host has internet access. Packages are pulled from the Vizrt registry.
Prepare offline installation	Run on a host that *does* have internet, to produce a bundle for an air-gapped install.
Offline installation	Run on the air-gapped target host with the bundle produced above.

Additional workflows that are not full installations:

Workflow	When to use
Backup / Restore	Take or restore a backup of an existing installation.
Cluster context	Switch the installer between multiple managed clusters.
Settings	Change installer-host settings (registry, paths, etc.)

4.1.3 Cluster Topologies

You can install VAM in one of two topologies:

- **Single-node**: one machine runs everything. Recommended for evaluation and small productions.
- **Multi-node**: one or more control-plane nodes plus worker nodes, for redundancy and capacity. Recommended for production.

The installer supports both Hyper-V (the default on Windows) and VMware vSphere for provisioning the nodes. You can also point the installer at an existing Kubernetes cluster instead of letting it create one. See Prerequisites → Deployment target for what each option requires, and the VMware / vSphere deployment guide for the vSphere path.

4.1.4 Where to Go Next

- **First-time install**: read the Prerequisites, then follow Running the installer.
- **VMware vSphere target**: see the VMware / vSphere deployment guide.
- **Your own Kubernetes cluster**: see Existing / native Kubernetes cluster.
- **Air-gapped / no internet**: see Offline (air-gapped) installation.

- **Existing install, day-to-day operation**: head to the Configuration Service guide.
- **Upgrade an existing install**: see Upgrading.

4.2 Prerequisites

Check every item below **before** running the installer. The installer performs an automated prerequisite scan on its first screen, but fixing prerequisites after the fact is much slower than getting them right up front.

The installer needs **Administrator** rights and changes the host: it enables Hyper-V, creates and **deletes** virtual machines and virtual switches, runs a rootful Podman machine on WSL 2 (issuing `wsl --shutdown`, which stops every WSL distribution on the host), **formats** the disks you assign to cluster storage, and installs command-line tooling. Use a dedicated host for production. See Installer Overview for the full list.

4.2.1 Installer Host

The machine on which you run `vamctl.exe`.

Requirement	Minimum
OS	Windows Server 2022 or Windows 11 (Pro / Enterprise)
CPU	4 cores
RAM	8 GB
Disk	50 GB free on the system drive for installer state, logs, offline bundles
Privileges	An account with **local Administrator** rights
Network	Outbound HTTPS (443) to the Vizrt registry for online installs
Virtualization	Hardware virtualization enabled in BIOS/UEFI — required by both Hyper-V and WSL 2
WSL 2	Available and set as the default version. The installer runs Podman on it.

The installer host can be the same machine that runs VAM (for a single-node install on Hyper-V), or a separate management workstation (for VMware vSphere or remote Hyper-V deployments).

Podman and WSL 2

Every installation needs a working Podman machine, including a plain online install onto Hyper-V. The installer uses Podman to **build the Talos boot images** that provision the cluster nodes, and to **stage offline bundles**. Without it, the install cannot proceed past preparation.

The installer manages the machine for you: it creates one if none exists, sets it **rootful** with **user-mode networking**, sizes it, and starts it. You do not configure Podman by hand. What you must provide is the host support it depends on:

- WSL 2 installed, current, and the default version (`wsl --status`).
- Hardware virtualization enabled — WSL 2 needs it just as Hyper-V does.
- Enough headroom for the machine on top of Windows itself. On WSL 2 the machine's memory and CPU come from `%UserProfile%\.wslconfig` , not from Podman.

> ⚠ Podman on WSL 2 is the component that most often breaks after a Windows or WSL update. If preparation fails on the **Podman** row, see Troubleshooting the Installer.

4.2.2 Cluster Nodes

VAM runs as a set of containerised services on a **Kubernetes cluster**. The nodes are the machines — physical or virtual — that make up that cluster. The requirements below are **per node**.

A cluster has two kinds of node:

- **Control-plane nodes** run Kubernetes itself (the API server, scheduler and the etcd datastore). For a highly-available cluster you need an **odd** number so etcd keeps quorum — the installer offers **1** (evaluation) or **3** (production).
- **Worker nodes** run the VAM services and hold the persistent storage for media, indexes and databases.

A typical resilient production layout is therefore **3 control-plane + 3 or more worker nodes**. A single-node install collapses all roles onto one machine and is intended for evaluation and small productions.

Single-Node Evaluation

Requirement	Minimum
CPU	8 cores
RAM	32 GB
Disk	500 GB SSD
OS	Provisioned by the installer (Talos Linux)

Multi-Node Production

Role	CPU	RAM	Disk	Count
Control plane	4 cores	16 GB	200 GB SSD	3 (for HA)

Role	CPU	RAM	Disk	Count
Worker	16 cores	64 GB	1 TB+ SSD	3 or more

Production sizing depends heavily on the size and volume of media your installation handles; the numbers above are a starting point. Ask your Vizrt representative for a sizing review before ordering hardware.

How the Installer Splits Disk per Node

You size each node with **one number** — its total disk. The installer divides that total into the two kinds of disk a node actually gets:

- a **system disk** (the Talos OS / ephemeral disk) that holds the container images plus every non-media persistent volume — OpenSearch, the Keycloak database, SurrealDB, RabbitMQ and friends all live here, because that storage class is backed by the node's own filesystem;
- one or more **object-storage drives**, separate disks that hold the S3 object store (media, proxies, artifacts).

The system portion is fixed by the installed software, so it does **not** grow with the total. Everything you add on top of it becomes asset storage — a bigger node means a bigger S3 store, not a bigger operating system.

Node role	Total minimum	Total recommended	Split at the recommended total
Single-node (control plane runs workloads)	130 GB	200 GB	120 GB system + 80 GB object storage
Control plane in a cluster that has workers	40 GB	60 GB	all system (no object storage)
Worker	130 GB	200 GB	120 GB system + 80 GB object storage

The minimums come from the floors of the two parts: 80 GB of system disk and 50 GB per object-storage drive. At exactly 130 GB the split lands on both floors at once; above that the system disk holds at 120 GB (or at whatever `TalosSystemDiskSize` is set to in appsettings) and the rest goes to the asset store.

A control plane only stays small when the cluster actually has worker nodes — it is tainted and runs no application pods, and it receives no object-storage drive. In a **single-node** install there are no workers, so the control plane is schedulable, carries the whole stack and holds the asset store.

The wizard accepts anything at or above the minimums and warns — without blocking — below the recommended totals.

On Hyper-V these are *dynamically expanding* VHDX files: a 120 GB system disk only occupies as much host disk as it actually contains, so sizing generously up front costs nothing until the data exists. Sizing too small does cost you, because existing VHDs are never resized by a later run of the installer.

> *The cluster nodes are created and managed for you by the installer — they run **Talos Linux**, a minimal immutable Kubernetes OS, so you do not install or patch a node operating system by hand. The only exception is the **bring-your-own Kubernetes** option below, where you supply the cluster yourself.*

4.2.3 Deployment Target

The installer can either **provision the cluster nodes for you** (on Hyper-V or VMware vSphere) or **deploy into a Kubernetes cluster you already run**. Pick the option that matches your infrastructure.

Hyper-V (Windows) — Single Host

The simplest option: the installer creates the Talos VMs on a single Windows Hyper-V host.

- Hyper-V role enabled on the host (the installer can enable it for you if you accept a reboot).
- A virtual switch already configured with network connectivity to the rest of your environment (internal or external vSwitch).
- Enough free RAM and disk on the host to satisfy the node sizing above — for a single-node evaluation, budget the full node spec plus headroom for the host OS.

Hyper-V Host Cluster (HA Production)

For a highly-available production deployment, run the VAM cluster across a **failover cluster of at least 3 Hyper-V hosts**, so the loss of one physical host does not take VAM down. Recommended **per Hyper-V host**:

Component	Recommended
CPU	16-core (or dual 8-core) with VT-x/AMD-V + EPT/NPT (for example, AMD EPYC 7302P / Intel Xeon Silver 4210)
Memory	64 GB RAM (to comfortably allocate 8 GB+ per VM)
System disk	512 GB NVMe SSD (Hyper-V + host OS)
VM storage	2 TB+ NVMe / SAS SSD (local or shared) for VM data
Networking	2× 10 GbE NICs (VM traffic + storage replication)
Host OS	Windows Server 2025 (Core) or equivalent

For shared / cluster storage sizing on a Hyper-V failover cluster, ask your Vizrt representative — it depends on your media volume and retention.

VMware vSphere

The installer can also provision the Talos nodes on VMware vSphere.

- **vCenter 7.0 or later** (ESXi 6.7 Update 2+ / hardware version `vmx-15`), reachable from the installer host.
- A datacenter, host/cluster, datastore and a **DHCP-enabled** network / port group for the VMs.
- vCenter **content libraries** available (vCenter 6.5+).
- A vSphere account with permission to create VMs, attach storage and manage networking. The exact privilege set is in the VMware / vSphere deployment guide.

- A pre-uploaded Talos Linux OVA, or internet access from the installer host to download it.

See the VMware / vSphere deployment guide for the full walkthrough.

Bring Your Own Kubernetes Cluster

Instead of letting the installer create a cluster, you can point it at an **existing Kubernetes cluster** (on-prem or cloud). The installer then skips all VM / OS provisioning and only deploys the VAM Helm packages. That cluster must provide:

Requirement	Notes
Kubernetes version	A currently-supported version — confirm the minimum with your Vizrt representative.
Admin `kubeconfig`	A kubeconfig with cluster-admin rights; the installer binds to its **current context**.
Dedicated to VAM	The cluster (or at least its VAM namespaces) must not be shared with unrelated workloads.
Persistent storage	A default `StorageClass`, or block devices for DirectPV, sized per the worker guidance above.
Ingress / exposure	A way to reach the cluster from clients — a `LoadBalancer` service or NodePort ingress.
Node capacity	Enough total CPU / RAM / disk to satisfy the worker sizing above.

For the full prerequisites and the step-by-step install into your own cluster — storage (DirectPV), certificates (including **AD Certificate Services**), ingress and identity — see the Existing / native Kubernetes cluster guide.

4.2.4 Network

Item	Notes
Static IP addresses for the cluster nodes	DHCP reservations by MAC address are acceptable as long as they are stable.
DNS A record for the VAM endpoint	For example, `vam.example.com` — required for TLS certificates and sign-in.
DNS server reachable from the nodes	For in-cluster and external name resolution.

Item	Notes
NTP / time source reachable from the nodes	Talos and Kubernetes require accurate time (for example, `time.cloudflare.com` or internal NTP).
Preconfigured virtual switch (Hyper-V / vSphere)	Internal or external, allowing VM-to-VM communication.
HTTPS (443) outbound to `registry.vizrt.cloud` (online)	Or your internal mirror, for online installs. Not required for offline installs.
Open ports between nodes	The installer lists and verifies them on the prerequisite screen.
Firewall: 443 (HTTPS), 6443 (Kubernetes API)	Inbound to the control-plane node from clients that use VAM and the config UI.

Enterprise identity & PKI (optional). *Larger deployments often integrate with* ***Microsoft Entra ID*** *for sign-in (see Identity provider setup) and an internal certificate authority such as* ***AD Certificate Services*** *for TLS. Neither is required to install VAM — the installer can self-generate certificates and use the bundled Keycloak — but they are the recommended path for production.*

4.2.5 Licensing

VAM requires a valid **WIBU CodeMeter license**. Two delivery options:

- **Soft license container** (CmActLicense file): sent by Vizrt as a `.WibuCmRaU` file. You load it via the installer's license step or directly through CodeMeter Control Center.
- **Hardware dongle** (CmDongle): plugged into a USB port on a host that the cluster can reach.

Have your license file or dongle available before starting the installer.

4.2.6 Identity Provider (for the Configuration Service)

VAM uses OpenID Connect (OIDC) for sign-in. Two options:

- **Use the bundled Keycloak**: the installer can deploy a Keycloak instance for you. Easiest for evaluations; nothing to prepare in advance.
- **Use an existing OIDC provider**: Microsoft Entra ID (Azure AD), Okta, Auth0, or your own Keycloak.

Provider	Preparation needed before install
Bundled Keycloak	None — the installer deploys and configures it.
Microsoft Entra ID	Register an app, expose an API, define roles, add a secret. See the Identity provider setup guide.

Provider	Preparation needed before install
Okta	Create an OIDC app and groups; register redirect URIs. See Other providers.
Auth0	Create a web application and a `policy`-claim action. See Other providers.
External Keycloak	Create a realm client and roles; register redirect URIs. See Other providers.

For an external provider, have ready before you start:

- The provider **host / issuer URL** (for Entra: `https://login.microsoftonline.com` plus your **tenant ID**).
- A **client ID** and **client secret** created for VAM.
- The redirect URIs to register, including `https://<your-vam-host>/config/auth/callback` and `https://<your-vam-host>/auth/callback`.
- For Entra specifically: the app roles defined (the `vam` for normal VAM usage and `vam-admin` for the cluster administration is mandatory) and admin consent granted.

Set up the external provider first. *Completing it before you launch the installer turns the install into a single pass. Two guides cover it:* OIDC Authentication Provider Configuration *walks through the Entra ID app registrations step by step, and the* Identity provider setup guide *covers every supported provider plus the values the installer asks for.*

You can change the identity provider later through Platform Settings, but it is easier to configure it during installation.

The next page, OIDC Authentication Provider Configuration, is the detailed walkthrough for preparing Microsoft Entra ID. Skip it if you are using the bundled Keycloak.

4.2.7 Before You Start the Installer

A short pre-flight checklist:

- [ ] Installer host meets the requirements above.
- [ ] You have chosen a deployment target (Hyper-V, VMware vSphere, or an existing Kubernetes cluster) and its hardware / hypervisor capacity is ready.
- [ ] Static IPs, DNS record and a reachable NTP source allocated.
- [ ] WIBU license available.
- [ ] You know which OIDC option you will use — and, for an external provider, you have completed the Identity provider setup and have the client ID, secret and (for Entra) tenant ID to hand.
- [ ] You have a backup destination in mind (file share, S3, or Azure Blob) — you will configure it after install.

Once these are in place, continue to Running the installer.

4.3 OIDC Authentication Provider Configuration

4.3.1 General

This page details configuration steps for OIDC providers.

The OpenID Connect Protocol (OIDC) is used to authenticate users. It can, in theory, be configured to integrate with any OIDC compliant identity provider.

The following providers have been tested and are supported:

- Microsoft Entra ID (formerly Azure Active Directory)
- Vizrt SSO (Keycloak)

4.3.2 Microsoft Entra ID

The Entra ID application configuration consists of two parts: an App Registration for the technical details of the application, and an Enterprise Application that defines user and group access and their role assignments, as well as other aspects of how the app integrates with your tenant.

Detailed Instructions

Complete instructions to set up the Entra ID App Registrations.

Introduction

This guide walks you through the steps to create and configure two App Registrations in your Microsoft Entra ID (Azure AD) tenant: "Vizrt Back-Ends" and "Vizrt MSE". These configurations allow your applications to use Entra ID for authentication and authorization.

When creating the registrations manually, both `appId` (Application Client ID) and `id` (Object ID) are auto-generated by Azure AD.

Step 1: Create the "Vizrt Back-Ends" App Registration

1. **Navigate to Azure Portal:** Go to the Azure Portal and log in.
    a. Select your tenant that should be configured on the top right (user details) under **Switch directory**.
2. **Access Entra ID:** Search for and select **Azure Active Directory**.
3. **Go to App Registrations:** In the left-hand menu, under **Manage**, select **App registrations**.
4. **Create New Registration:** Click on **+ New registration**.
    - **Name:** `Vizrt Back-Ends`
    - **Supported account types:** Select **Accounts in this organizational directory only (Single tenant)**.
    - Leave the **Redirect URI (optional)** section blank for now.
    - Click **Register**.

> ⚠ The registration name is free-form. This guide uses "Vizrt Back-Ends" as the example throughout; use whatever fits your naming convention.

If an application needs a client secret, create it in the same registration under **Certificates & secrets > Client secrets > New client secret**. Give it a description and an expiry that matches your rotation policy, then copy the value immediately — Entra shows it only once. Sign-in breaks when a secret lapses, so track the expiry date.

Step 2: Define App Roles for "Vizrt Back-Ends"

1. In the "Vizrt Back-Ends" App Registration, navigate to **App roles** under **Manage**.
2. Click on **+ Create app role**. Create the following roles. Ensure the settings match exactly.
    a. For each role, click **Apply** after entering the details.

Some role values of VAM-Messaging require the `{APPCLIENTID}` to be set to the application (client) ID as prefix. This can be retrieved from the app registration status page.

Role Display Name	Allowed member types	Value	Description
VAM-Storage Read/Write	User, Application	`readwrite`	VAM-Storage Read/Write
VAM-SERVICE	User, Application	`vam`	VAM-SERVICE
VAM-SERVICE Admin	User	`vam-admin`	VAM-SERVICE Admin
VAM-Messaging Write All	User, Application	`{APPCLIENTID}.write:*/*/*`	VAM-Messaging Write All
VAM-Messaging Read All	User, Application	`{APPCLIENTID}.read:*/*/*`	VAM-Messaging Read All
VAM-Messaging Configure All Vhosts	User, Application	`{APPCLIENTID}.configure:*/*`	RabbitMQ Configure All Vhosts
pilot-admin	User, Application	`pilot-admin`	Has administrator access to Viz Pilot Settings and all apps

Role Display Name	Allowed member types	Value	Description
graphic-designer	User, Application	`graphic-designer`	Can use Template Builder
pilot-editor	User, Application	`pilot-editor`	Can use Pilot Edge
pilot-journalist	User, Application	`pilot-journalist`	Can use Pilot Edge
pilot-mse	Application	`pilot-mse`	Used by the Media Sequencer Engine (MSE) to connect to Viz Pilot
VAM-Storage Admin UI	User	`consoleAdmin`	MinIO Admin UI

⚠ The VAM-Messaging roles authorize service-to-service messaging. The storage and message-broker administration consoles are not part of this configuration, so no browser sign-in is set up for them.

1. **Go to API permissions:** In the left-hand menu, under **Manage**, select **API permissions**.
2. **Add Microsoft Graph Permissions:**
   - Click **+ Add a permission**.
   - Select **Microsoft Graph**.
   - Choose **Delegated permissions**.
   - Search for and select the following permissions: `email`, `offline_access`, `openid`, `profile`, and `User.Read`.
   - Click **Add permissions**.
3. **Add Vizrt Back-Ends Permissions:**
   - Click **+ Add a permission**.
   - Select the **My APIs** tab.
   - Find and click on **Vizrt Back-Ends**.
   - Choose **Application permissions**.
   - Search for and select the following permissions (the display names are in parentheses for easier identification):
     - `{APPCLIENTID}.configure:*/*` (VAM-Messaging Configure All Vhosts)
     - `{APPCLIENTID}.read:*/*/*` (VAM-Messaging Read All)
     - `{APPCLIENTID}.write:*/*/*` (VAM-Messaging Write All)
     - `readwrite` (VAM-Storage Read/Write)
     - `vam` (VAM-SERVICE)

- `pilot-mse` (pilot-mse)
- Click **Add permissions**.

4. **Grant Admin Consent:** After adding all permissions, you will likely need to grant admin consent. Click the **Grant admin consent** button and then **Yes**. Ensure the status of all permissions changes to **Granted**.

API Permission	Permission Name	Type	Admin Consent Required
Microsoft Graph	`email`	Delegated	No
Microsoft Graph	`offline_access`	Delegated	No
Microsoft Graph	`openid`	Delegated	No
Microsoft Graph	`profile`	Delegated	No
Microsoft Graph	`User.Read`	Delegated	No
Vizrt Back-Ends	`{APPCLIENTID}.configure:*/*`	Application	Yes
Vizrt Back-Ends	`{APPCLIENTID}.read:*/*/*`	Application	Yes
Vizrt Back-Ends	`{APPCLIENTID}.write:*/*/*`	Application	Yes
Vizrt Back-Ends	`readwrite` (VAM-Storage Read/Write)	Application	Yes
Vizrt Back-Ends	`vam` (VAM-SERVICE)	Application	Yes
Vizrt Back-Ends	`pilot-mse` (pilot-mse)	Application	Yes

Step 3: Expose API for "Vizrt Back-Ends"

1. In the "Vizrt Back-Ends" App Registration, navigate to **Expose an API** under **Manage**.
2. Click **Set** next to **Application ID URI**.
3. Accept the default `api://{APPCLIENTID}` and click **Save**.

Step 4: Configure Redirect URLs for "Vizrt Back-Ends"

1. Open the newly created "Vizrt Back-Ends" App Registration.
2. In the left-hand menu, under **Manage**, select **Authentication**.
3. **SPA Redirect URIs:**
    - In the **Single-page application (SPA)** section, click **Add a URI**.
    - Add one URI per web application, replacing `FQDN` with the host name of your VAM installation:
        - `https://FQDN:30443/config/`
        - `https://FQDN:30443/pilot/pcs-admin/`
        - `https://FQDN:30443/pilot/pilotedge/`
        - `https://FQDN:30443/pilot/template-builder/`
        - `https://FQDN:30443/pilot/data-server-config/`
    - Click **Save**.

> ⚠ Entra matches a redirect URI exactly, including the trailing slash. Add one set of URIs for every host name users reach the installation by. Port `30443` is the gateway reached directly; omit the port when the installation sits behind a load balancer that terminates on 443.

Step 5: Create the "Vizrt MSE" App Registration

1. Go to Azure Active Directory (Entra ID) then **App registrations**.
2. Click **+ New registration**.
    - **Name:** `Vizrt MSE`
    - **Supported account types:** Select **Accounts in this organizational directory only (Single tenant)**.
    - Leave the **Redirect URI (optional)** section blank for now.
    - Click **Register**.

Step 6: Request API Permissions for "Vizrt MSE"

1. In the "Vizrt MSE" App Registration, navigate to **API permissions** under **Manage**.
2. Click on **+ Add a permission**.
3. Select the **My APIs** tab.
4. Find and click on **Vizrt Back-Ends**.
5. Under **Application permissions**, select the following roles:

Role Display Name	Role ID	Type
VAM-Storage Read/Write	`<role-id>`	Role
VAM-SERVICE	`<role-id>`	Role
pilot-mse	`<role-id>`	Role

Role Display Name	Role ID	Type
pilot-admin	`<role-id>`	Role
graphic-designer	`<role-id>`	Role
pilot-editor	`<role-id>`	Role
pilot-journalist	`<role-id>`	Role

⚠ Role IDs are generated when a role is created, so they differ in every tenant and every app registration. Select the roles by display name in the portal; read the IDs from the **App roles** blade of "Vizrt Back-Ends", or from its **Manifest**, if you need them.

1. Click **Add permissions**.
2. Click the **Grant admin consent** button and then **Yes**. Ensure the status of all permissions changes to **Granted**.

Step 7: Assign Roles to Users/Groups for "Vizrt Back-Ends"

1. Go to Azure Active Directory (Entra ID) then **Enterprise applications**.
2. Find the **Vizrt Back-Ends** application.
3. In the left-hand menu, under **Manage**, select **Users and groups**.
4. Click on **+ Add user/group**.
5. Select the users or groups you want to grant access to.
6. Under **Select a role**, choose the appropriate role or roles you defined in Step 2.
7. Click **Assign**. Repeat this for all necessary user/group and role assignments.

Step 8: Enforce V2 JWT Tokens

1. Go to Azure Active Directory (Entra ID) then **App registrations**.
2. Open both the "Vizrt Back-Ends" and "Vizrt MSE" App Registrations.
3. In the left-hand menu, under **Manage**, select **Manifest**.
4. Find the `"api"` section in the JSON manifest.
5. Ensure the `"requestedAccessTokenVersion"` property is set to `2`. If it is not present, or is set to `null` or `1`, change it to `2` and click **Save**.

Completing these steps configures your Entra ID App Registrations. Remember to configure your actual applications with their respective Application (client) IDs, any necessary client secrets, and the correct scope and audience information for authentication and authorization.

4.4 Running the Installer

This is the end-to-end walkthrough for a first-time online installation. Offline and hybrid flows are noted where they diverge.

4.4.1 Launching `vamctl`

1. On the installer host, run `vamctl.exe` as Administrator.
   - From a Vizrt distribution: double-click the installer executable.
   - From an existing install: it is available at `C:\Program Files\Vizrt\vamctl\vamctl.exe`.
2. A console window opens briefly, then your default browser launches and shows the wizard at `http://127.0.0.1:<port>/`. The wizard opens on its first step, **Prepare Installer**.

> The installer hosts its own local web server on a loopback port. Leave the console window open for the duration of the install. If you close it, the wizard disconnects.

If the browser does not open automatically, copy the URL from the console window into a browser manually.

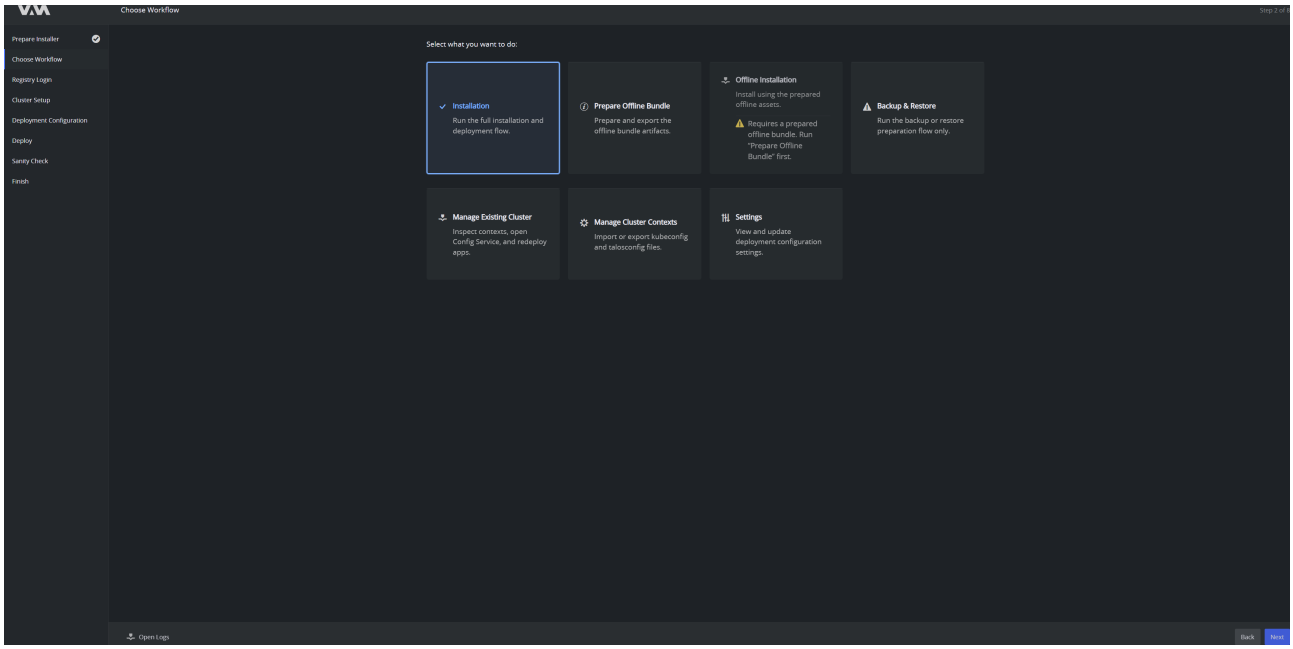
The wizard tracks its own progress in the left sidebar, and the header shows the current step. The **total** number of steps depends on the workflow you choose — creating a new cluster adds the Cluster Setup sub-steps — so this guide refers to screens by name rather than number.

4.4.2 Choose a Workflow

Select what you want to do. For a fresh installation, choose **Installation**.

Workflow	Use it to
Installation	Run the full installation and deployment flow.
Prepare Offline Bundle	Prepare and export the offline bundle artifacts.
Offline Installation	Install using a prepared offline bundle. Run **Prepare Offline Bundle** first.
Backup & Restore	Run the backup or restore preparation flow only.
Manage Existing Cluster	Inspect contexts, open the Configuration Service and redeploy apps.
Manage Cluster Contexts	Import or export kubeconfig and talosconfig files.
Settings	View and update deployment configuration settings.

Click **Next**.



4.4.3 Prepare Installer

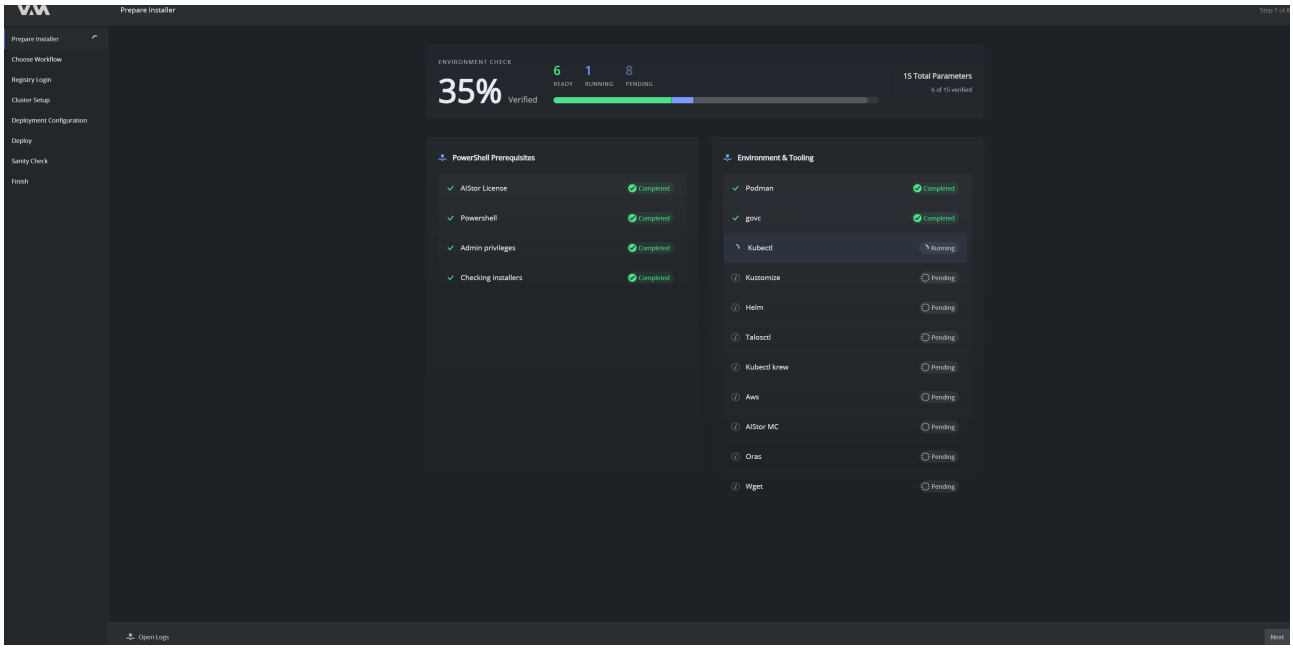
The first step runs an **environment check**. It verifies the host and then downloads and verifies the command-line tools the installer needs. The header shows overall progress and how many parameters are verified.

PowerShell Prerequisites covers the host itself: the AIStor license, the PowerShell version, administrator privileges and the installers already present on the machine.

Environment & Tooling verifies each tool in turn — Podman, govc, Kubectl, Kustomize, Helm, Talosctl, Kubectl krew, Aws, AIStor MC, Oras and Wget. Each row reports **Completed**, **Running**, or **Pending**.

Podman is the row that matters most, and the one most likely to fail. The installer uses it to build the Talos boot images that provision the cluster nodes and to stage offline bundles, so it is required for every install — including a plain online install onto Hyper-V. It runs as a rootful Podman machine on WSL 2, which the installer configures itself.

Wait until every row reports Completed, then click **Next**. If a row fails, use **Open Logs** at the bottom of the window to read the reason. For Podman failures see Troubleshooting the Installer.



4.4.4 Registry Login

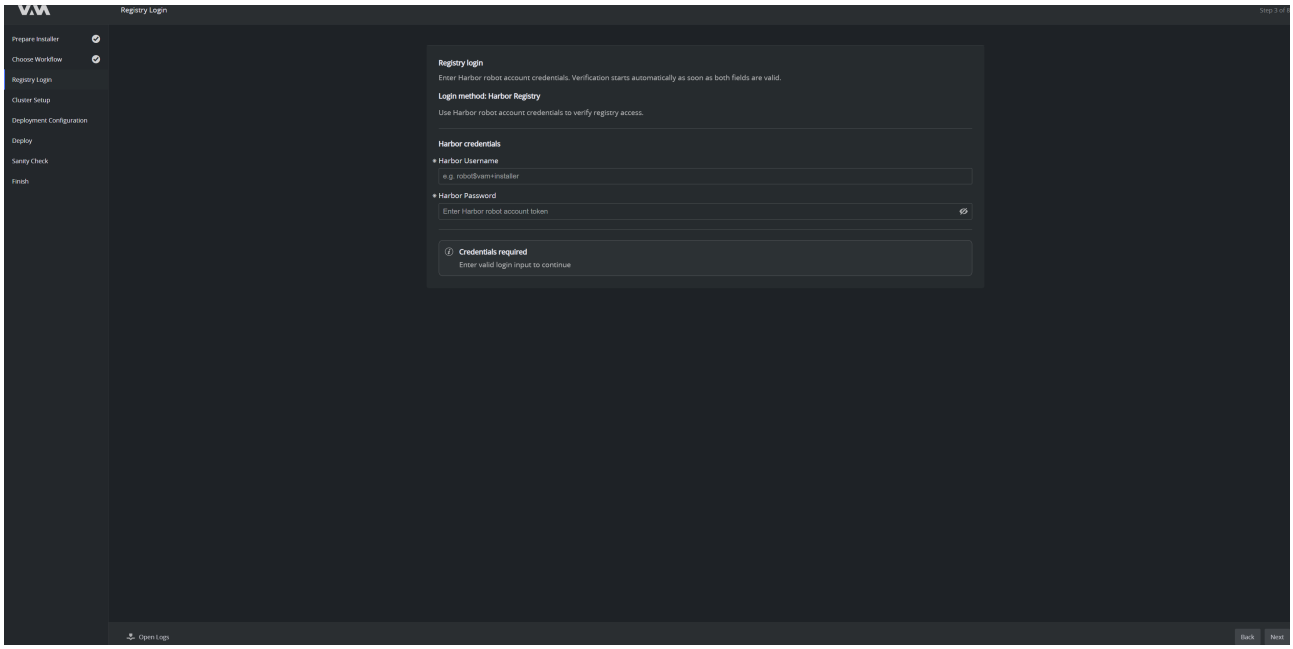
The installer pulls VAM container images and Helm packages from the Harbor registry, so it needs credentials for a **Harbor robot account**.

1. In **Harbor Username**, type the robot account name, in the form *robot$vam+installer*.
2. In **Harbor Password**, type the robot account token.

Verification starts automatically as soon as both fields are valid — there is no separate button to press. Until then the panel reports *Credentials required*. When verification succeeds, click **Next**.

If verification fails, see Troubleshooting → Registry access.

For an offline install, use the bundle you produced earlier instead. See Offline (Air-Gapped) Installation for the full prepare-transfer-install flow.



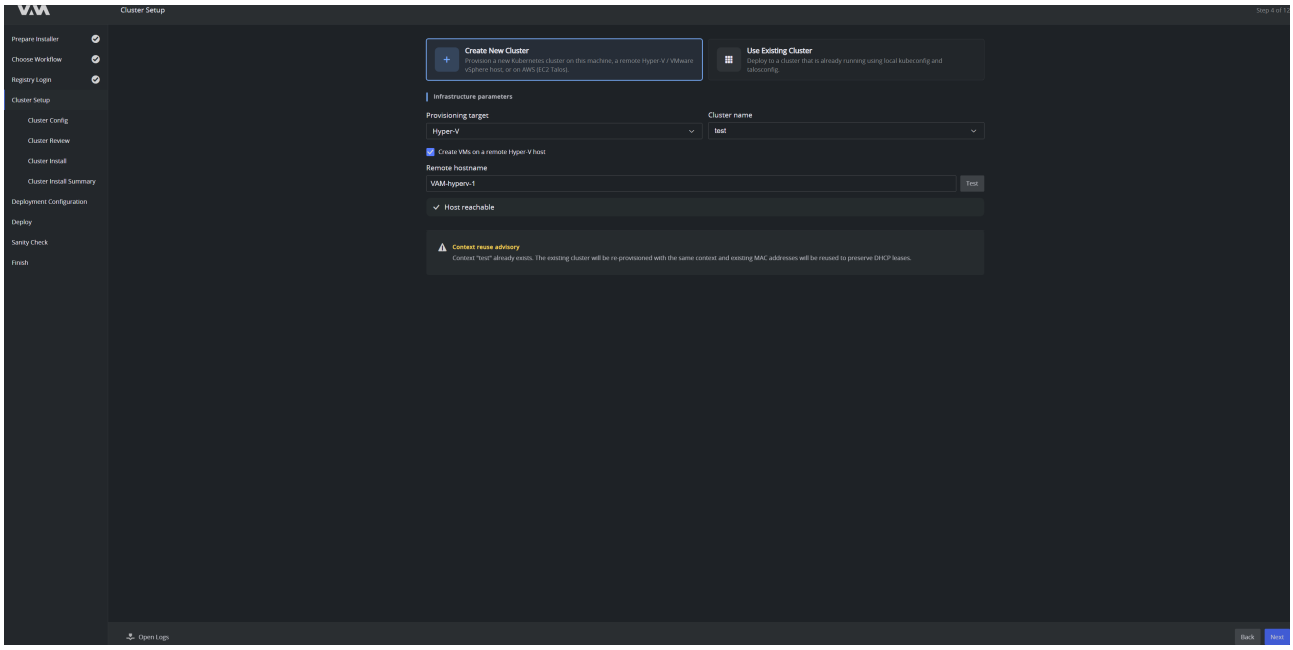
4.4.5 Cluster Setup

> ⚠ The **Installation** workflow always installs the current versions from the registry — there is no version-selection step. To pin versions, use the **Prepare Offline Bundle** workflow, which lets you choose a version per package. See Offline (Air-Gapped) Installation.

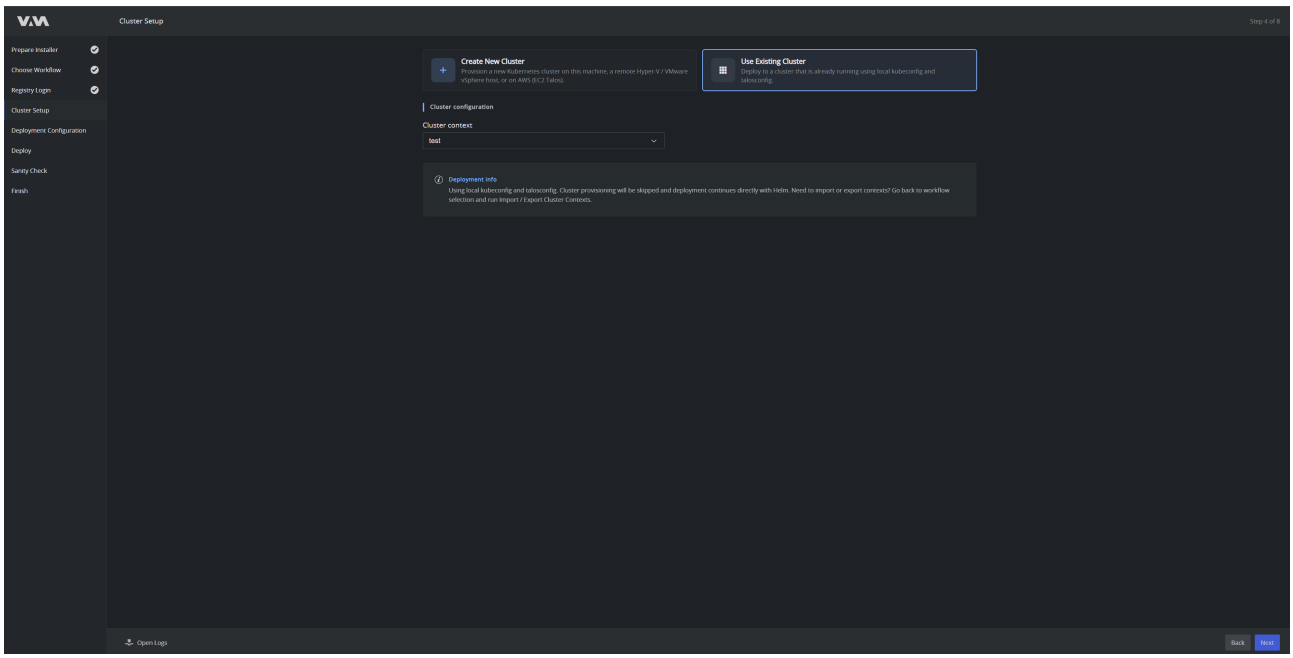
Choose whether the installer builds a cluster or deploys onto one you already run.

Create New Cluster provisions a new Kubernetes cluster on this machine, on a remote Hyper-V or VMware vSphere host, or on AWS (EC2 Talos). Under **Infrastructure parameters**:

- **Provisioning target**: the platform that hosts the nodes, for example *Hyper-V*.
- **Cluster name**: the name of the cluster context. If the name already exists, the wizard shows a *Context reuse advisory*: the existing cluster is re-provisioned with the same context, and its MAC addresses are reused so DHCP leases survive.
- **Create VMs on a remote Hyper-V host**: select this to provision on another host, type its name in **Remote hostname**, then click **Test**. The panel reports *Host reachable* when the connection succeeds.



Use Existing Cluster deploys to a cluster that is already running, using the local kubeconfig and talosconfig. Select the **Cluster context** to deploy into. Cluster provisioning is skipped and deployment continues directly with Helm. To import or export contexts first, go back to the workflow selection and run **Manage Cluster Contexts**.



For **VMware vSphere**, follow the dedicated VMware vSphere Deployment guide for the connection details and required vCenter permissions.

For an existing cluster, the cluster must be empty or dedicated to VAM — the installer does not share a namespace with unrelated workloads. See Existing Kubernetes Cluster for the full prerequisites and flow (storage, certificates, ingress, identity).

4.4.6 Cluster Config

Define the cluster shape. A **Resource Forecast** panel on the right shows the estimated memory overhead for the current configuration, so you can see the cost of a change as you make it.

Topology

In **Deployment model**, choose:

- **Single-node**: fastest path for evaluations.
- **Multi-node**: distributes workloads across dedicated control-plane and worker nodes for high availability.

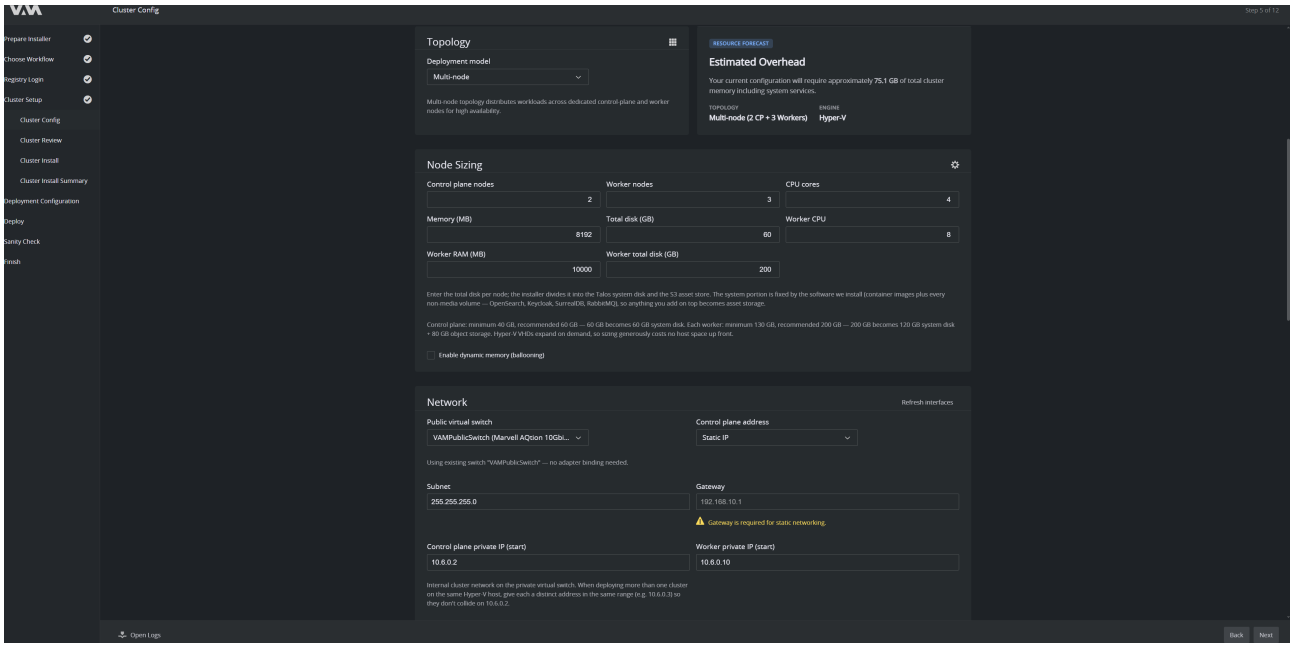
Node Sizing

- **Control plane nodes** and **Worker nodes**: how many of each.
- **CPU cores**, **Memory (MB)**, **Total disk (GB)**: per control-plane node.
- **Worker CPU**, **Worker RAM (MB)**, **Worker total disk (GB)** — per worker node.
- **Enable dynamic memory (ballooning)**: leave cleared unless the host is memory-constrained.

Total disk is the whole disk per node. The installer divides it into the Talos system disk and the S3 asset store; the system portion is fixed, so anything you add on top becomes asset storage. A control plane needs a minimum of 40 GB (60 GB recommended). Each worker needs a minimum of 130 GB (200 GB recommended, which becomes a 120 GB system disk plus 80 GB object storage). Hyper-V disks expand on demand, so sizing generously costs no host space up front.

Network

- **Public virtual switch**: the switch the nodes attach to.
- **Control plane address**: *Static IP* or DHCP.
- **Subnet** and **Gateway**: a gateway is required for static networking.
- **Control plane private IP (start)** and **Worker private IP (start)** — the internal cluster network on the private virtual switch. When deploying more than one cluster on the same Hyper-V host, give each a distinct address in the same range so they do not collide.
- **Hyper-V destination path**: where the node disks are stored. Use **Browse folders** to pick it.



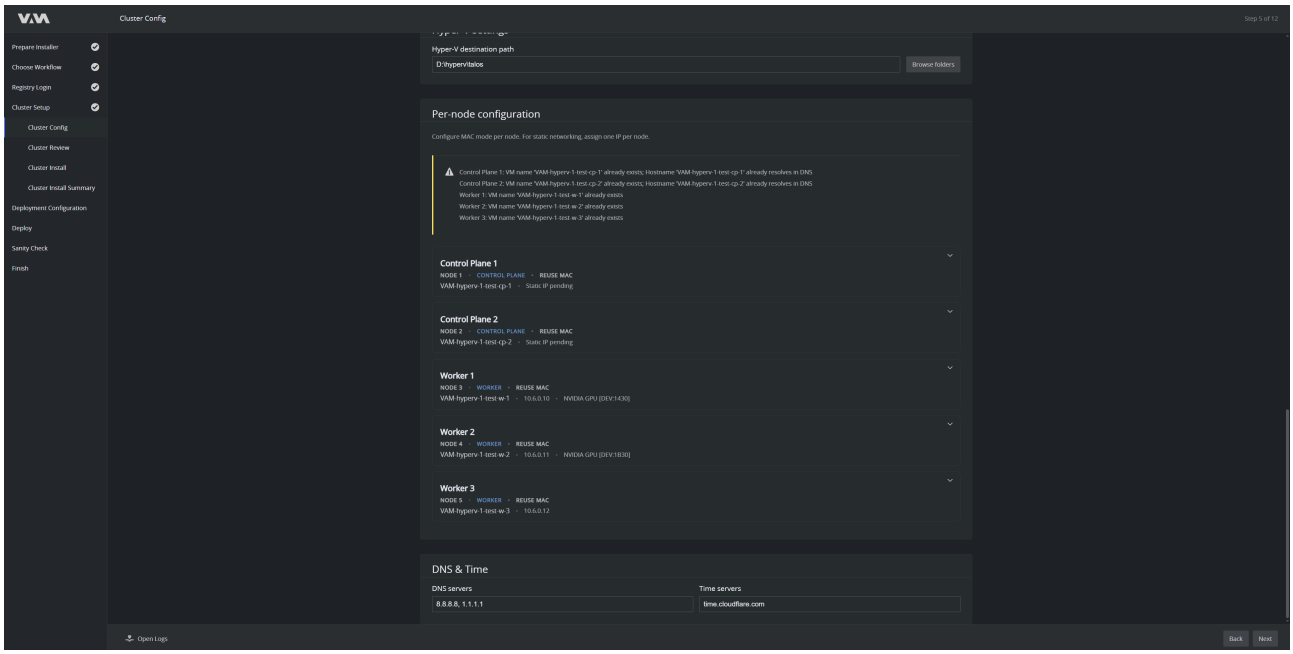
Per-Node Configuration

Each node appears as a card showing its role, hostname, MAC mode and assigned address. Expand a card to set the MAC mode for that node, and to assign a **GPU** to a worker from the devices the installer found on the host. For static networking, assign one IP per node.

If a node's VM name or DNS record already exists, the wizard lists the conflict above the cards.

DNS and Time

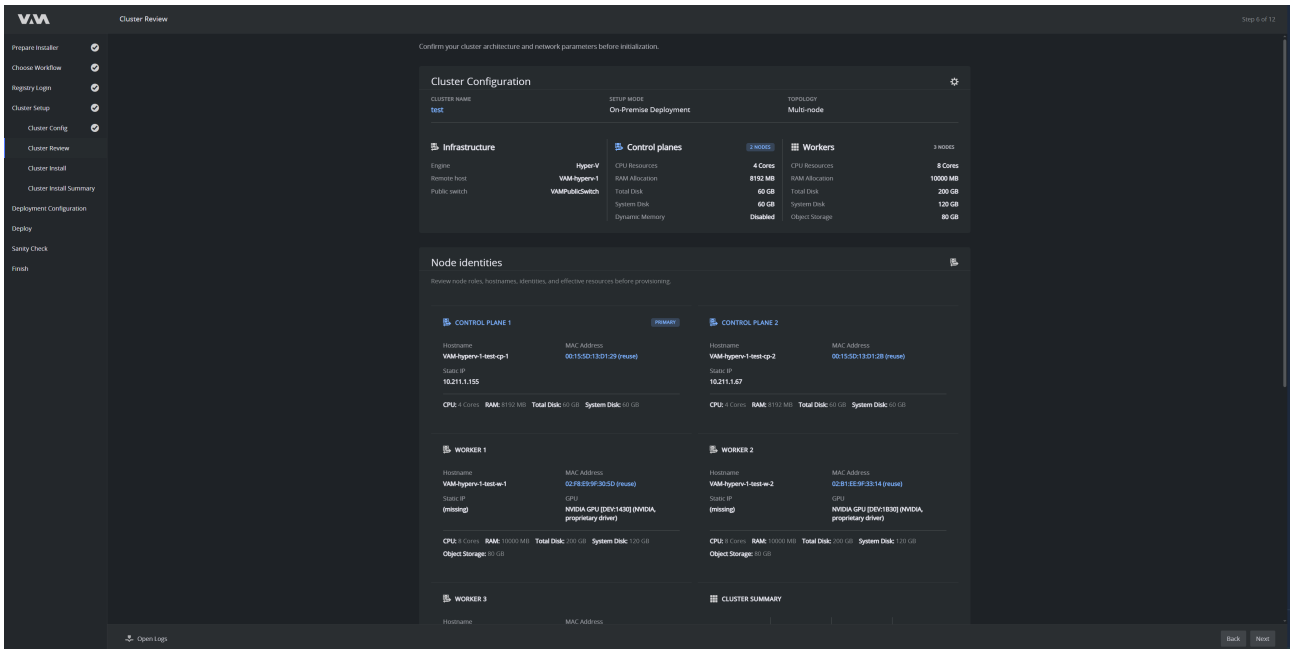
Set **DNS servers** and **Time servers** for the cluster nodes.



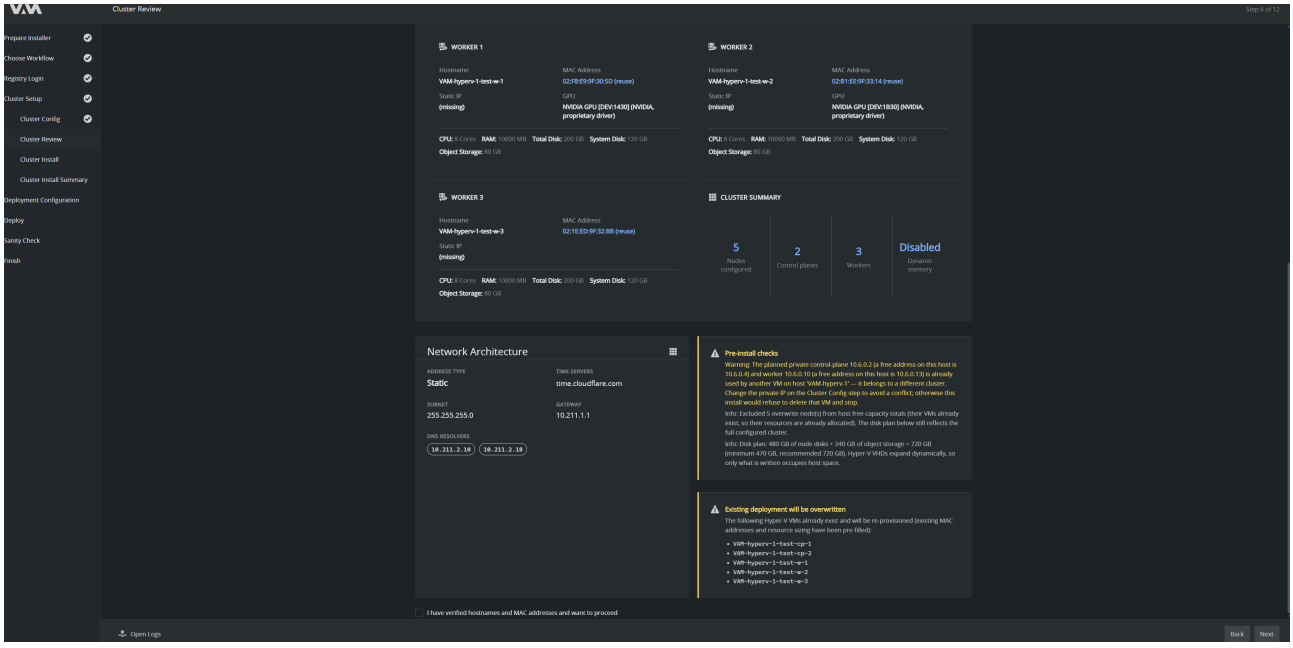
4.4.7 Cluster Review

Confirm the cluster architecture and network parameters before initialization. **Cluster Configuration** summarises the cluster name, setup mode, topology, infrastructure and the resources allocated to control planes and workers. **Node identities** lists each node with its hostname, MAC address, static IP, GPU and effective resources.

Read this screen carefully. Changing the topology after this step requires reinstalling the cluster.



Further down, **Network Architecture** shows the address type, subnet, gateway, DNS resolvers and time servers. **Pre-install checks** lists any warnings — for example a planned private IP that is already used by a VM belonging to another cluster, which you must resolve on the Cluster Config step. If VMs from a previous install exist, **Existing deployment will be overwritten** names each one that is re-provisioned.



Select **I have verified hostnames and MAC addresses and want to proceed**, then click **Next**.

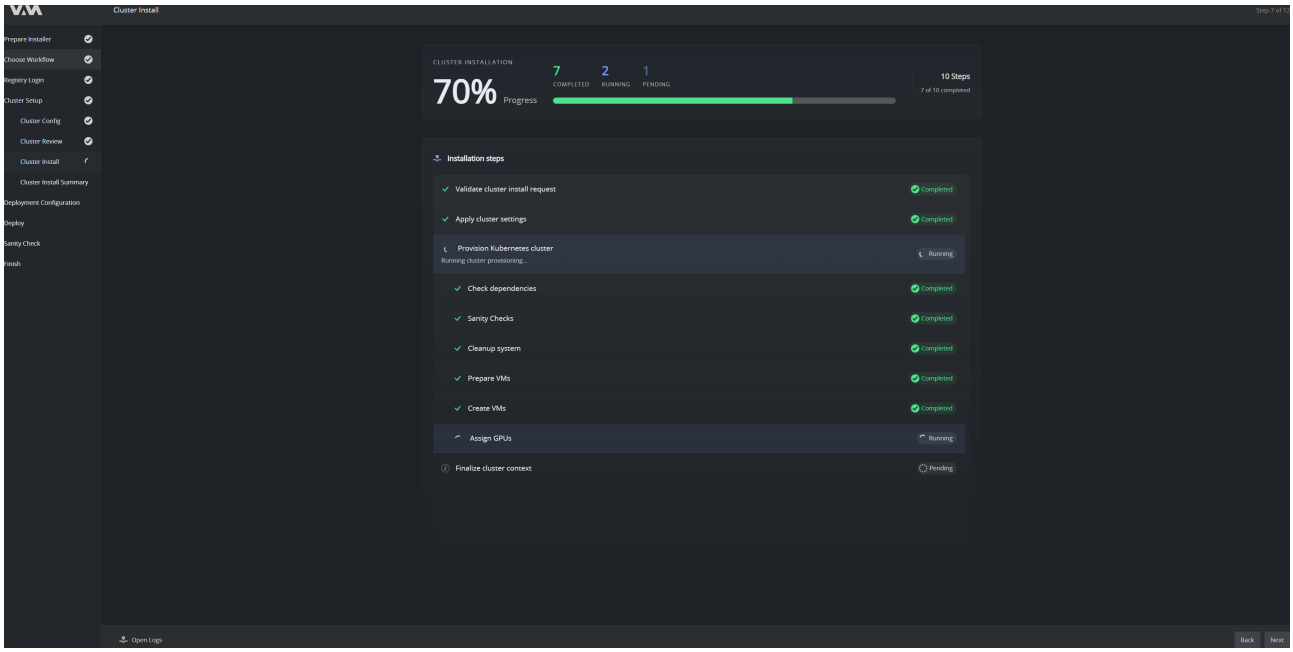
4.4.8 Cluster Install

The installer:

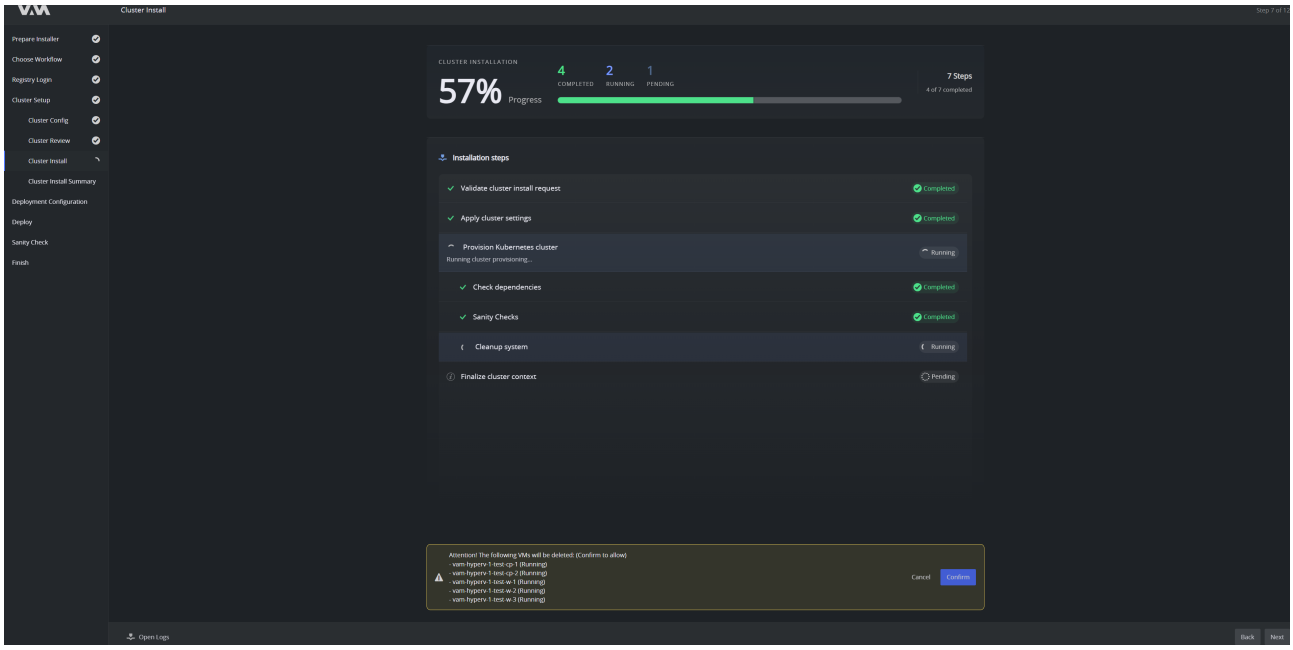
1. Provisions the node VMs (or boots existing nodes via Talos).
2. Installs Kubernetes.
3. Joins worker nodes to the control plane.
4. Verifies that the cluster is healthy.

Progress is shown step-by-step with live log output. If a step fails, the wizard pauses and shows the error; expand the step to read the log, then either **Retry** or **Cancel** to go back.

This typically takes 10–25 minutes for a single-node install and longer for multi-node. **Do not close the wizard window** during this step.



If the target host already has VMs from a previous install, the wizard pauses and asks you to confirm that they may be deleted and re-provisioned. Nothing is removed until you click **Confirm**.



4.4.9 Cluster Install Summary

When installation finishes, the wizard shows a summary of the cluster it built: each node with its hostname, role and address and the result of the final health verification.

Check that every node reports as expected and that the node count matches what you configured. If a node is missing or unhealthy, use **Open Logs** to read the installation log before continuing — it is easier to correct the cluster now than after VAM is deployed onto it.

The cluster context is also written to your kubeconfig at this point, so you can inspect the cluster with `kubectl` from the installer host.

Click **Continue**.

4.4.10 Deployment Configuration

This step has two views, shown as **1 Infrastructure** and **2 Deployment Options** at the top of the page.

View 1 — Infrastructure

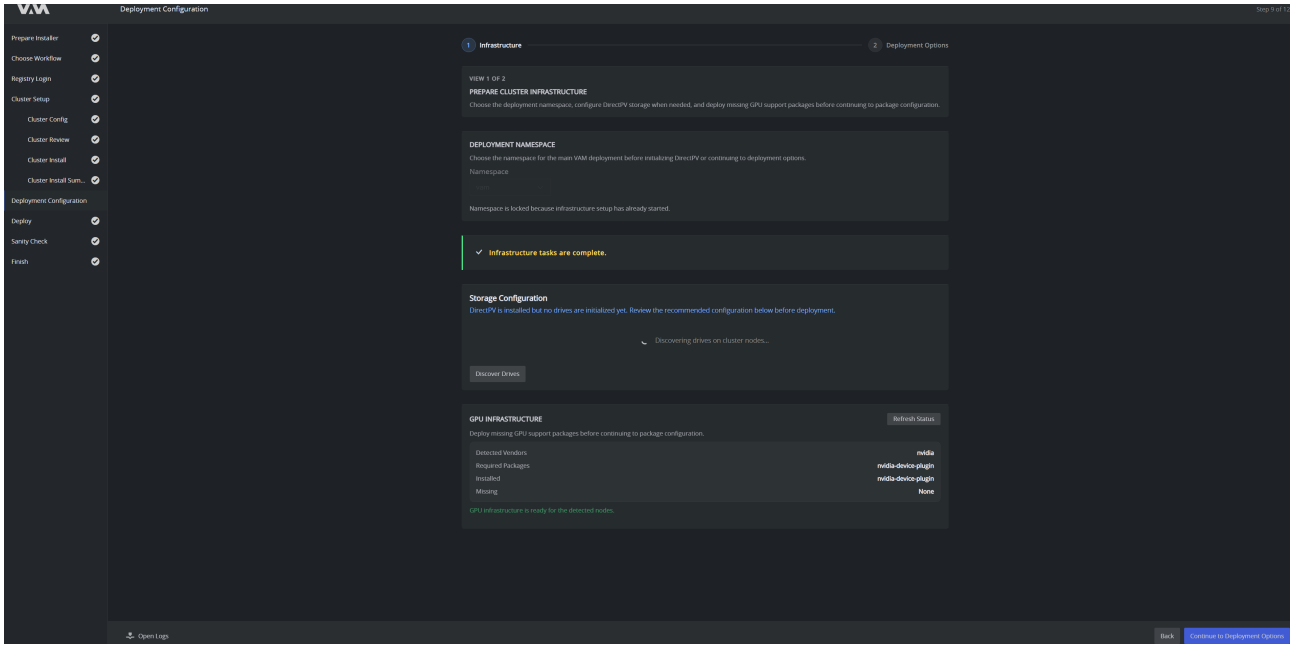
Prepare the cluster infrastructure: choose the deployment namespace, configure DirectPV storage and deploy any missing GPU support packages.

- **Deployment namespace**: the namespace for the main VAM deployment. Once infrastructure setup has started the namespace is locked, and the field explains why.
- **Storage Configuration**: DirectPV is installed with the platform, but its drives are not initialized yet. The **Recommended Configuration** panel proposes a layout from the drives it finds and reports usable capacity, raw capacity, storage efficiency and how many drive and node failures the layout survives. Read the

warnings before accepting: they call out HDD-only drives, drives with an existing filesystem that **are formatted** and single-node layouts that have no node-failure redundancy.

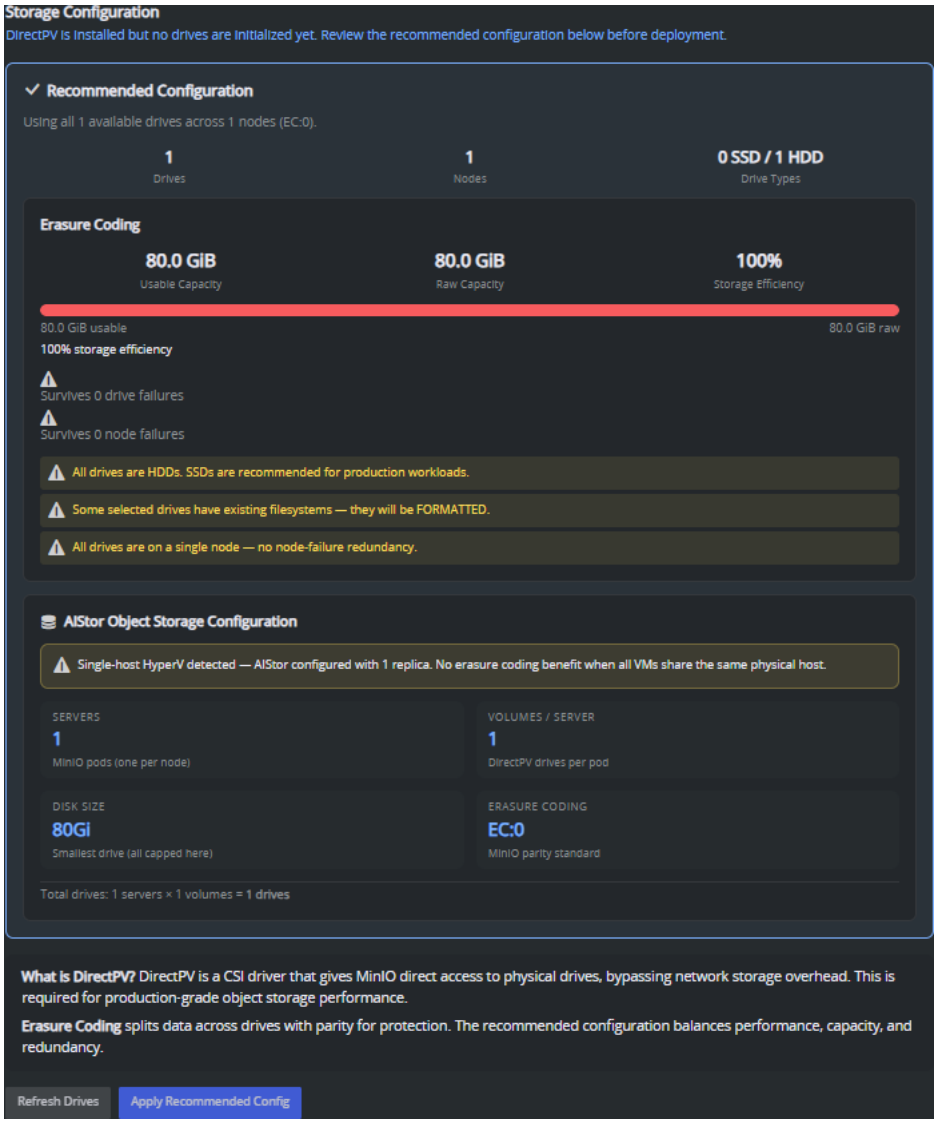
- **AIStor Object Storage Configuration**: derived from the same drives: servers, volumes per server, disk size and the erasure coding level.
- **GPU Infrastructure**: lists detected vendors and the required, installed and missing GPU packages. Use **Refresh Status** after installing one.

On first entry the panel discovers the drives on the cluster nodes; use **Discover Drives** if it has not started, or **Refresh Drives** to re-scan later.



When discovery finishes, the **Recommended Configuration** panel appears. Review it, then click **Apply Recommended Config** to accept the layout.

Applying a configuration formats any selected drive that already has a filesystem. Confirm the drive list before you apply it.



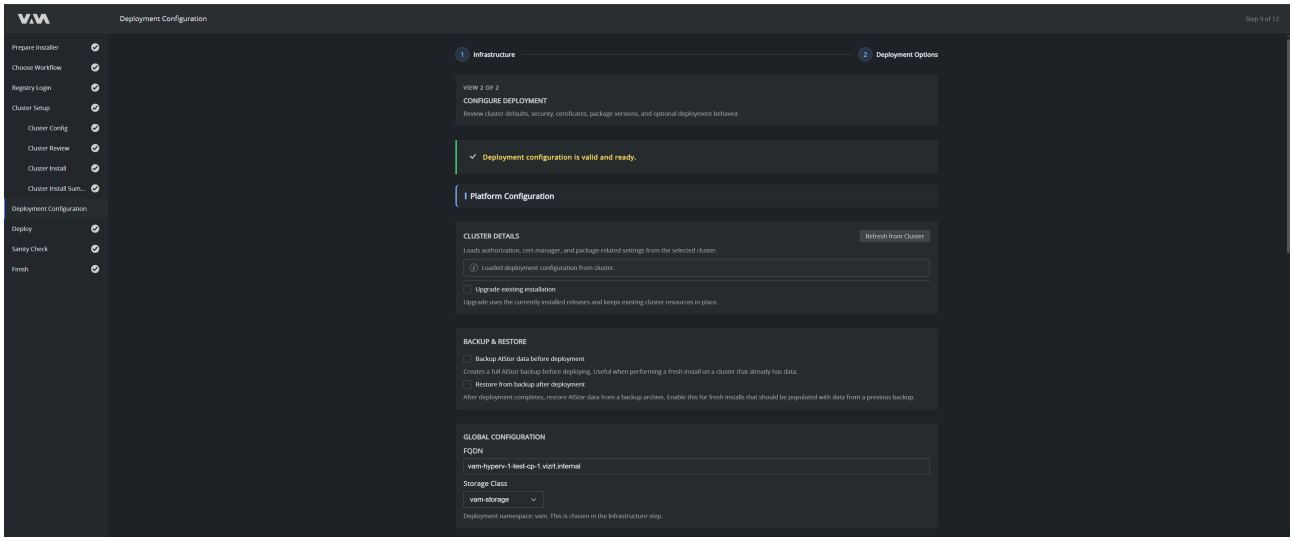
Click **Continue to Deployment Options**.

View 2 — Deployment Options

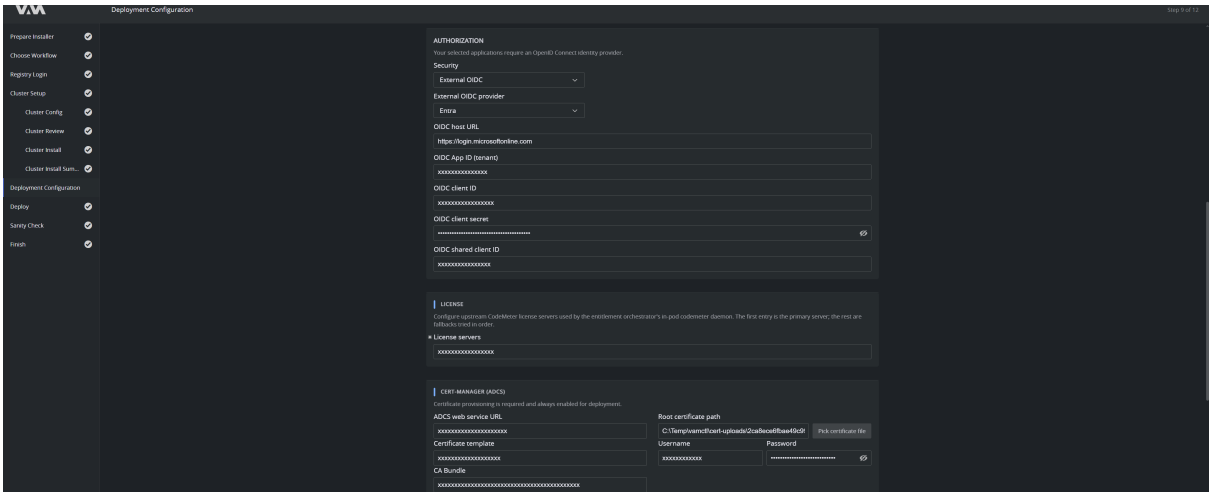
Review cluster defaults, security, certificates, package versions and optional deployment behavior. The banner at the top reports whether the configuration is valid and ready.

- **Cluster Details**: the installer loads the existing deployment configuration from the selected cluster. Use **Refresh from Cluster** to reload it. Select **Upgrade existing installation** to keep the currently installed releases and existing cluster resources in place.
- **Backup & Restore**: optionally **back up AIStor data before deployment** (useful for a fresh install onto a cluster that already holds data), and optionally **restore from a backup after deployment**.

- **Global Configuration**: the **FQDN** clients use to reach the platform, and the **Storage Class** for persistent volumes.



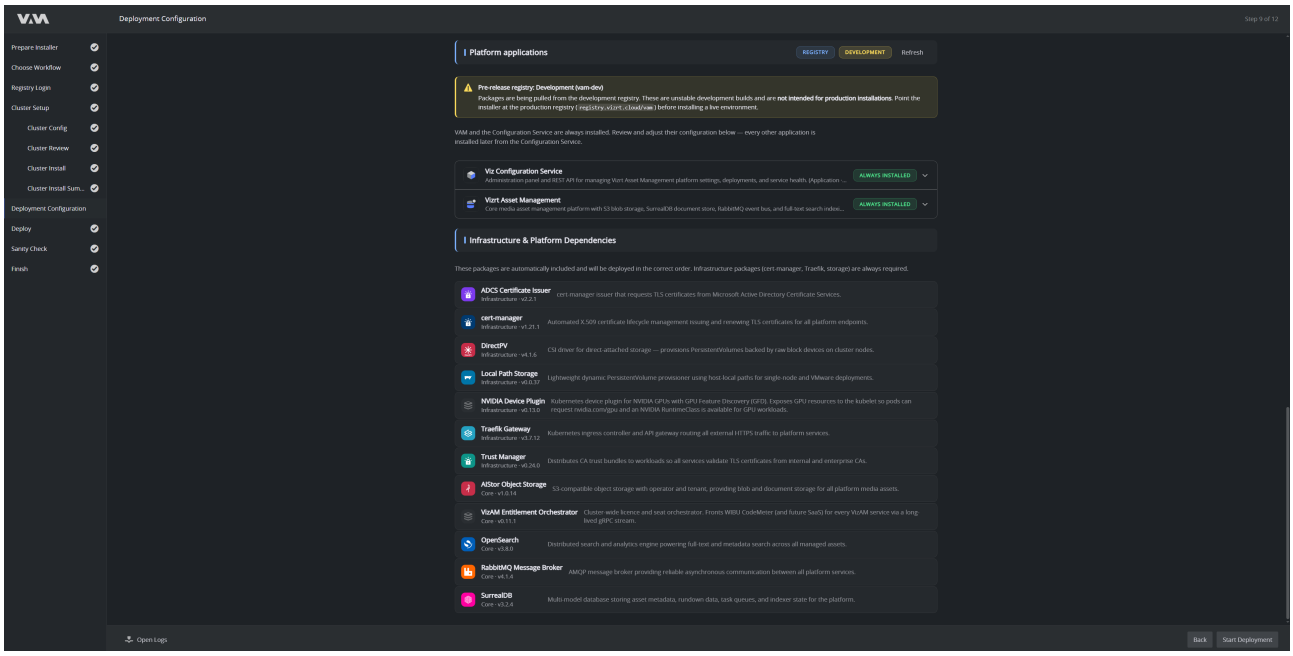
- **Authorization**: the identity provider. Choose the **Security** mode and, for **External OIDC**, the provider and its OIDC host URL, application (tenant) ID, client ID, client secret and shared client ID. Take these values from Prerequisites.
- **License**: the upstream CodeMeter **license servers** used by the entitlement orchestrator. The first entry is the primary server; the rest are tried in order as fallbacks.
- **Cert-Manager (ADCS)**: certificate provisioning is required and always enabled. Supply the ADCS web service URL, the root certificate path, the certificate template, the account credentials allowed to request certificates and the CA bundle.



- **Platform applications**: VAM and the Configuration Service are always installed. Every other application is installed later from the Configuration Service.

- **Infrastructure & Platform Dependencies**: the packages that are included automatically and deployed in the correct order, with their versions. Infrastructure packages (cert-manager, Traefik, storage) are always required.

The selector above the application list switches between the **Registry** (production) and **Development** registries. Development builds are unstable and are not intended for production installations. Point the installer at the production registry before installing a live environment.



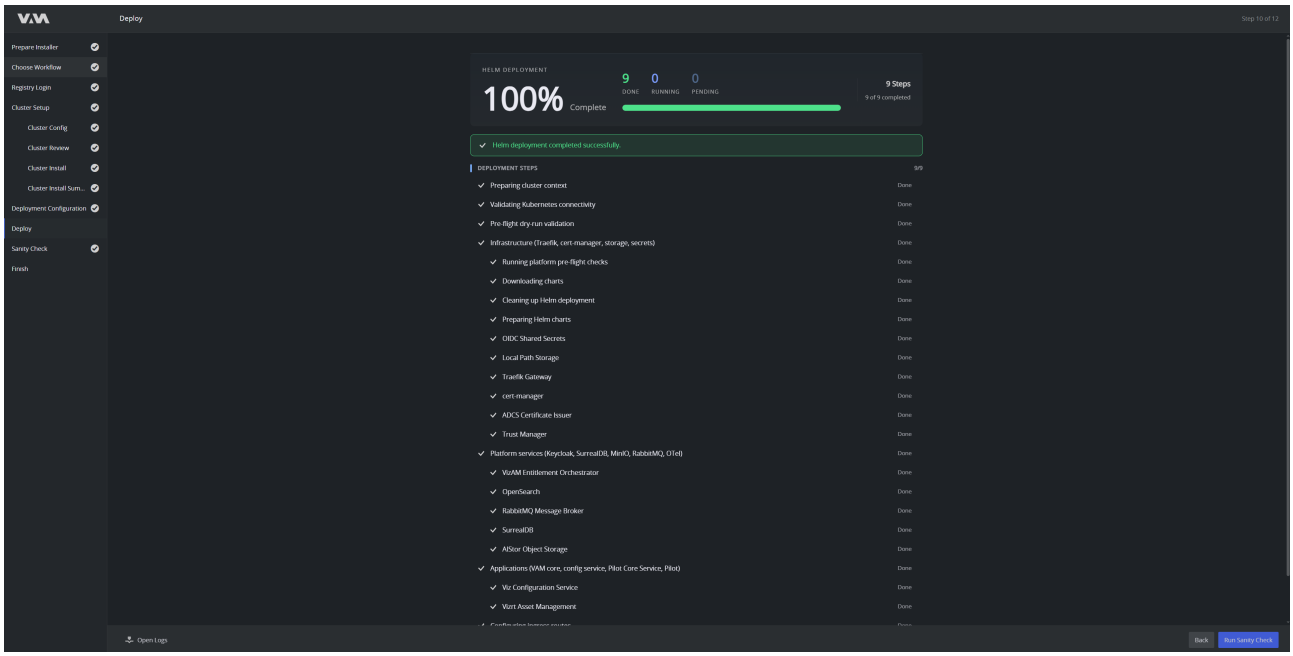
Click **Start Deployment**.

4.4.11 Deploy

The installer deploys VAM to the cluster with Helm. The header shows overall progress and the done / running / pending counts; the **Deployment steps** list expands into the individual releases.

The stages are: preparing the cluster context, validating Kubernetes connectivity, a pre-flight dry-run validation, infrastructure (Traefik, cert-manager, storage, secrets), platform services (Keycloak, SurrealDB, MinIO, RabbitMQ, OTel), the applications themselves and finally the ingress routes.

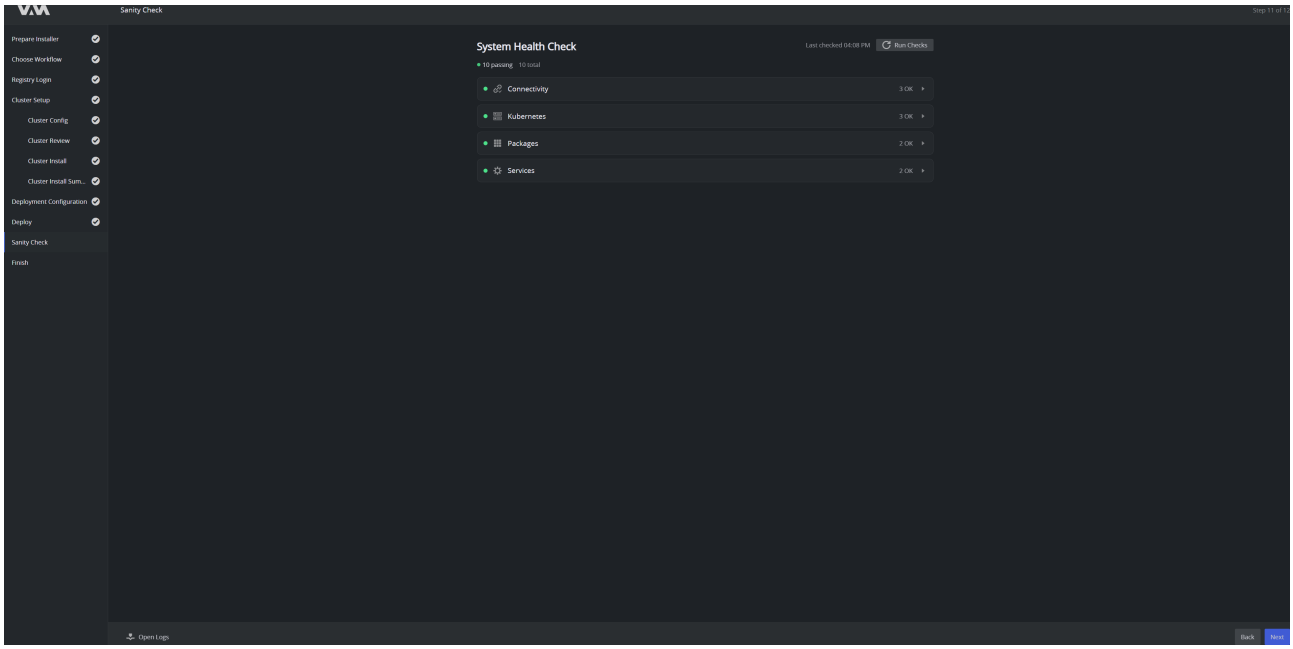
This can take 15–30 minutes depending on your registry bandwidth and node hardware. When it reports *Helm deployment completed successfully*, click **Run Sanity Check**.



4.4.12 Sanity Check

The **System Health Check** groups its results into **Connectivity**, **Kubernetes**, **Packages** and **Services** and reports how many checks pass out of the total. Expand a category to see the individual checks.

- **All passing**: click **Next** to finish.
- **A failing category**: expand it to read the diagnosis, fix the cause, then click **Run Checks** to test again.



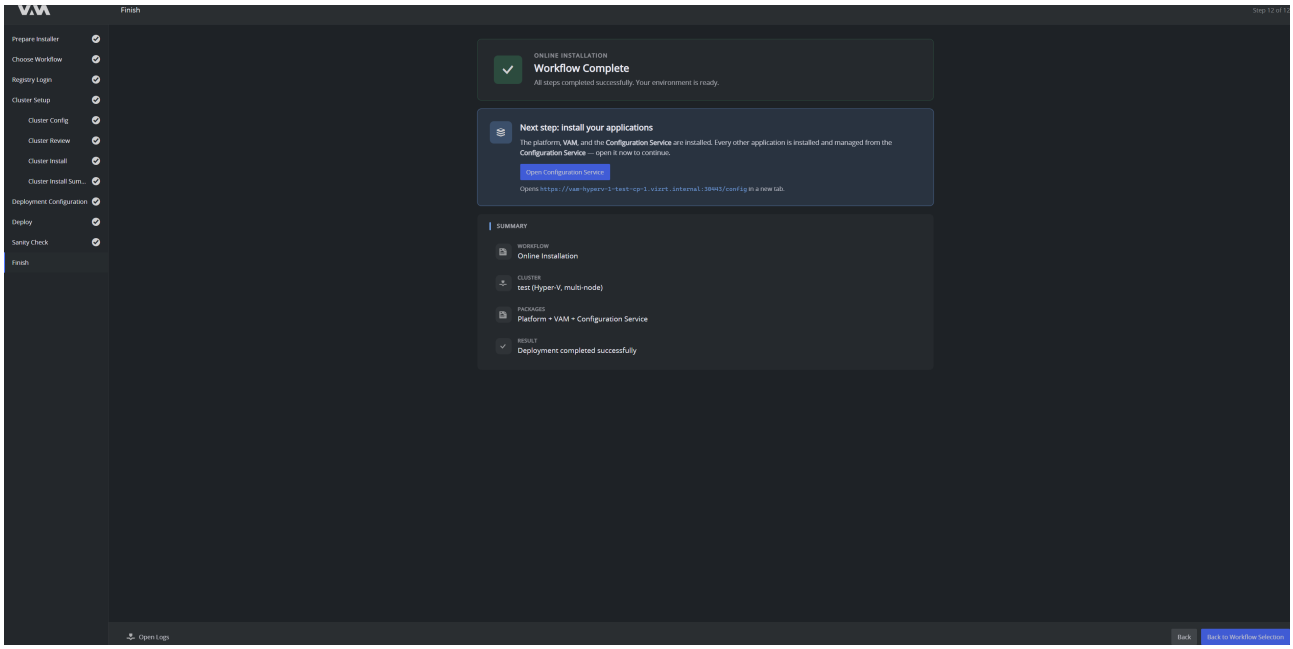
4.4.13 Finish

Workflow Complete confirms that every step succeeded and the environment is ready.

The platform, VAM and the Configuration Service are installed. Every other application is installed and managed from the Configuration Service, so the next step is to open it: click **Open Configuration Service**, which opens the URL shown beneath the button — the FQDN you set, on port 30443, at `/config` — in a new tab.

Summary records the workflow that ran, the cluster and its topology, the packages installed and the result.

Use **Open Logs** to save the install log if you need it for a support case, and **Back to Workflow Selection** to run another workflow, such as a backup.



4.4.14 Next Steps

- Post-install validation — recommended checks before you hand the installation over to users.
- Configuration Service → Signing in — first login walkthrough.

4.4.15 Resuming an Interrupted Install

If the installer was closed or the host rebooted, simply launch `vamctl.exe` again. The wizard resumes at the last completed step. If a long-running operation (cluster install or deployment) was in progress when interruption happened, the wizard offers to **reattach to the running operation** rather than restart it.

4.5 Post-Install Validation

After the installer reports success, do these checks before opening the installation to end users.

4.5.1 1. Sign in to the Configuration Service

Open the Configuration Service URL shown on the installer's Finish screen (typically `https://<your-vam-host>/config`).

- Sign in with your OIDC credentials. If you used the bundled Keycloak, the installer's Finish screen shows the initial admin username and password — change the password on first login.
- You should land on the **Packages** view. Confirm that the core VAM packages are listed and show **Healthy**.

If sign-in fails, see Troubleshooting → Cannot sign in.

4.5.2 2. Verify Cluster Health

From the Configuration Service, open **Cluster** in the navigation.

- All nodes should show **Ready**.
- The pod list should have **zero** pods in `CrashLoopBackOff`, `Pending` (longer than a few minutes), or `ImagePullBackOff`.
- The **Sanity check** page (in the navigation) should show all green.

4.5.3 3. Confirm Storage

In **Cluster → Storage**, confirm:

- The configured object storage backend (typically MinIO or an external S3) is reachable.
- Free capacity matches what you provisioned. A multi-node install should show capacity aggregated across worker nodes.

4.5.4 4. Configure a Backup Destination

Backups are **not** configured automatically. Before going to production:

1. Open **Backup** in the Configuration Service.
2. Click **Configure destination**.
3. Pick **Local file share**, **S3-compatible**, or **Azure Blob** and fill in the credentials.
4. Run **Test connection**.
5. Schedule a recurring backup (recommended: nightly).
6. Manually trigger a **first backup** and confirm it completes.

See Backup and Restore for full details.

4.5.5 5. Smoke-Test an End-User Workflow

If your distribution includes sample data:

1. Sign in to the VAM end-user UI (the public hostname without `/config`).
2. Ingest a small media file.
3. Confirm it appears in the asset list and the thumbnail generates.
4. Search for it.
5. Download / preview it.

If your distribution does not include sample data, pick a small representative file from your archive and repeat the steps above.

4.5.6 6. Save a Recovery Bundle

From the installer's main dashboard (re-launch `vamctl.exe` if needed), use **Export configuration** to save:

- The wizard answers used at install time.
- The cluster's `kubeconfig`.
- A copy of the installation log.

Store this bundle in a safe location (not on the installer host). You will need it if the installer host itself fails and you need to manage the cluster from another machine.

4.5.7 Ready for Users

If all six checks pass, your installation is ready. Hand the end-user URL to your users and proceed to day-2 operations via the Configuration Service.

4.6 Upgrading

VAM upgrades are performed with the same `vamctl` installer that did the original install. The installer runs upgrades against the existing cluster — there is no separate "upgrade tool".

> ***Warning**: take a backup immediately before any upgrade, and confirm that the backup succeeded.*

4.6.1 What an Upgrade Changes

A normal upgrade replaces the deployed VAM packages with newer versions. It does **not** by default:

- Upgrade Kubernetes itself.
- Reprovision cluster nodes.
- Change your platform settings (hostname, certificates, OIDC).

Kubernetes node upgrades are a separate operation; see Upgrading Kubernetes nodes below.

4.6.2 Online Upgrade

1. **Take a backup.** Confirm in **Configuration Service → Backup** that the latest backup is recent and succeeded.
2. On the installer host, launch `vamctl.exe`.
3. From the dashboard, choose **Upgrade VAM**.
4. The installer fetches the list of available versions from the package source. Select the target version.
5. The wizard shows a **change summary**: per package, the current version → target version.
6. Click **Start upgrade**.
7. The installer upgrades packages one at a time, in dependency order, with live progress.
8. After deployment, a **sanity check** runs automatically — same as on first install.

Typical duration: 10–30 minutes for a routine upgrade.

4.6.3 Offline Upgrade

The flow is identical to online except for the package source:

1. On a machine with internet access, run **Prepare offline installation** in the installer and select the target version. This produces an upgrade bundle.
2. Transfer the bundle to the upgrade host (USB, file share, etc.).
3. On the upgrade host, run **Upgrade VAM** and select **Offline bundle** as the source.

4.6.4 Rollback

If an upgrade fails or causes problems, restore from the backup you took in step 1:

1. Launch `vamctl.exe`.
2. From the dashboard, choose **Restore**.
3. Select the pre-upgrade backup.
4. Confirm the restore plan. The restore replaces the cluster contents.
5. Run the restore.

Full restore details are in Backup and Restore.

> *Rolling back individual packages (without restoring a full backup) is **not** supported through the UI. If a single package is causing issues, contact support.*

4.6.5 Upgrading Kubernetes Nodes

Kubernetes itself is upgraded separately, because it is disruptive (nodes are drained, restarted and rejoined one at a time):

1. Take a backup.
2. Launch `vamctl.exe`.
3. From the dashboard, choose **Cluster → Upgrade Kubernetes**.
4. Confirm the target Kubernetes version (the installer presents only versions compatible with your current VAM release).
5. The installer drains and upgrades each node in turn, waiting for the cluster to stabilise before moving on.

Single-node clusters experience downtime during this operation. Multi-node clusters with redundant control planes continue serving requests throughout.

4.6.6 After an Upgrade

Re-run the post-install validation checks. In particular, watch the cluster for the first 24 hours — most issues surface within minutes, but slow leaks or background-job failures can take longer.

4.7 Uninstalling

Uninstalling VAM removes the VAM application, the supporting services and (optionally) the Kubernetes cluster the installer created.

> ***Warning**: uninstall is destructive. All assets, configuration, and metadata stored on the cluster are deleted. Take a backup first if there is any chance you may want the data back.*

4.7.1 What Gets Removed

You can uninstall at three levels:

Level	What it removes
Application only	VAM packages and their data. Kubernetes cluster and `vamctl` itself stay in place.
Application + cluster	The above, plus the Kubernetes cluster (Hyper-V / VMware nodes are destroyed).
Everything	The above, plus `vamctl` from the installer host.

4.7.2 Procedure

1. **Take a backup** (if there is any data you may want later).
2. Launch `vamctl.exe` on the installer host.
3. From the dashboard, choose **Uninstall**.
4. Pick the level (see table above).
5. Type the confirmation phrase the installer shows you. This prevents accidental destructive runs.
6. Click **Uninstall**.
7. Wait for completion — typically a few minutes for application-only, longer if VMs must be destroyed.

4.7.3 Manual Cleanup

If the installer fails partway through uninstall, or if you need to clean up after the installer host itself was lost, contact support. A clean re-install over a half-removed installation is **not** recommended.

After a successful application-level uninstall, the cluster is empty but functional and you can run **Install VAM** again from the same installer to do a fresh deployment.

4.7.4 Removing the Installer Host

After an "Everything" uninstall, the installer is gone but a few artefacts remain on the host:

- **Configuration directory**: `%APPDATA%\Vizrt\vamctl\` — wizard answers, logs, exports.
- **Cached packages**: `%LOCALAPPDATA%\Vizrt\vamctl\cache\`.

- **Backup destinations** that pointed to local paths on this host.

Delete these manually if you want the host fully clean. None of them contain VAM end-user data; they are installer state only.

4.8 Troubleshooting the Installer

Common installer problems and how to resolve them. If your issue isn't covered here, export the installer logs (**Help → Export logs**) and contact support.

4.8.1 The Installer Browser Tab Never Opens

Symptom: you launched `vamctl.exe`, the console window is open, but no browser appears.

Resolve:

1. Look at the console window — it prints a line like `Listening on http://127.0.0.1:54321/`.
2. Open that URL in a browser manually.
3. If the URL refuses connections, an existing `vamctl` may be running. Open Task Manager, end any leftover `vamctl.exe` processes and relaunch.

4.8.2 "Another Operation Is Already Running"

Symptom: starting a long operation (install, deploy, backup) returns an error saying another operation is in flight.

Resolve: the installer permits only one long-running operation at a time. Either:

- Click **Reattach** to view the in-progress operation, or
- Wait for the current operation to finish, then retry.

If the previous operation has obviously hung (no progress for many minutes, no log output), use **Cancel** on the reattached view, then retry.

4.8.3 Prerequisite Check Fails: "Hyper-V Is Not Enabled"

Resolve:

- Click the row to expand. The installer offers to enable Hyper-V for you — accept, and reboot when prompted.
- Or enable it manually: **Server Manager → Add Roles and Features → Hyper-V**, then reboot.

4.8.4 Prerequisite Check Fails: Virtualization Is Disabled

Symptom: the installer (or Hyper-V) reports that hardware virtualization is not available.

Resolve:

1. In **Task Manager → Performance → CPU**, check that *Virtualization* shows **Enabled**.
2. If it is disabled, enable it in the host BIOS/UEFI — typically **Advanced → System Options → Virtualization Technology (VT-x)** and **Virtualization Technology for Directed I/O (VT-d)** — then reboot.
3. On a nested/virtualized installer host, ensure the outer hypervisor exposes virtualization to the VM.

4.8.5 Podman: Machine Will Not Start

Podman is required for **every** installation, not just offline ones: the installer uses it to build the Talos boot images that provision the cluster nodes, and to stage offline bundles. It runs as a **rootful Podman machine on WSL 2** with user-mode networking. A Windows update, a WSL update, or a reboot that left WSL half-started is the usual cause of this breaking, and the failure surfaces on the **Prepare Installer** step or when the cluster is created.

Symptom: the **Podman** row on Prepare Installer fails or hangs, or cluster creation reports that the Podman machine is unavailable.

Resolve, in this order — stop after the step that fixes it:

1. Check what the machine thinks it is doing:

```
podman machine list
```

2. If it is stopped, start it. If it reports that it is *already* running while the installer disagrees, stop it first and start it again:

```
podman machine stop
podman machine start
```

3. Reset WSL itself. This is the single most effective step after a Windows or WSL update:

```
wsl --shutdown
wsl --update
```

   Then start the machine again. Note that `wsl --shutdown` stops **every** WSL distribution on the host.

4. Confirm WSL 2 is the default and the kernel is current:

```
wsl --status
wsl --set-default-version 2
```

5. Reboot the host. A pending Windows update that has not finished installing keeps WSL in a broken state until it does.
6. If the machine is still unusable, recreate it. This discards the machine's contents, which is safe — the installer rebuilds what it needs:

```
podman machine stop
podman machine rm
```

   Re-run the installer, which recreates the machine with the settings it requires.
7. If preparation still fails on the **Podman** row, let the installer rebuild Podman for you. See Podman: Recovery Modes.

⚠ The installer needs the machine to be **rootful** and to use **user-mode networking**, and it sets both itself. If you created a Podman machine by hand, expect the installer to stop and restart it while applying those settings.

Do not make the Podman WSL distribution boot systemd. Adding `[boot] systemd=true` to `/etc/wsl.conf` inside `podman-machine-default` is widely repeated online as a fix for an unreachable Podman socket, and it breaks the Talos image build: under that workload the distribution terminates and drops the API socket part-way through the build. The Podman machine image starts its API socket without systemd, so the init system is never the cause of an unreachable socket and changing it never fixes one.

4.8.6 Podman: Recovery Modes

The **Podman** row on Prepare Installer is a health check, not just a test. It verifies the Podman installation, rewrites `containers.conf` when it is wrong, realigns the Podman system connection with the running machine, and then probes the Podman API. Much of what the sections above describe by hand, the installer therefore already attempts on its own.

When the check cannot reach Podman it writes a diagnostic block to the log — the WSL distributions, the Podman machines, the Podman system connections and the raw `podman info` error — followed by the remediation steps to try, in order. Read it through **Help → Export logs**.

Choosing the Recovery Mode

One setting, **Reinitialize Podman**, decides how far that check may go to repair a broken installation:

- **Off** (the default): non-destructive recovery only. The installer starts an existing machine, repairs `containers.conf` and the system connection, and then stops and reports if that is not enough.
- **On**: destructive recovery is authorized. Once the non-destructive steps fail, the installer works through a ladder and re-tests after each rung — reset the machine (stop, remove, fresh init, start), then `podman system reset --force`, then uninstall and reinstall Podman through Scoop followed by another reset.

Destructive recovery removes the Podman machine, and can uninstall and reinstall Podman itself. Everything held in the machine — images, containers and volumes — is lost. That is safe on a host where Podman serves only VAM, because the installer rebuilds what it needs, but not on a host where Podman also runs other workloads. Turn the setting off again once Podman is healthy, so that a transient failure is answered with a restart rather than a rebuild.

Set it in either place — it is the same value:

Where	How
Installer GUI	**Settings**, then **Reinitialize Podman**. It is a normal setting, not one hidden behind **Show advanced**.
Configuration file	`ReInitializePodman` in `%APPDATA%\Vizrt\vamctl\appsettings.json`.

With the setting on, re-run the **Prepare Installer** step. Recovery runs as part of the Podman check, so there is nothing else to start.

> ⚠ The last rung of the ladder is skipped when Podman was not installed by the installer, which does not uninstall software it does not own. The log says so: *Podman is externally installed; installer cannot perform managed Scoop reinstall automatically.* See Podman: Two Installations Clash.

4.8.7 Podman: Two Installations Clash

Symptom: `podman` behaves differently depending on which console window runs it, two console windows report different versions, the installer cannot see a machine that `podman machine list` shows, or the log contains:

```
Careful: program podman has been externally installed. Please uninstall
the program and let the installer install it!
```

Cause: the installer installs and manages Podman through Scoop, and recognizes its own installation by the `scoop\shims` folder in the resolved executable path. A Podman installed any other way — the MSI from the Podman releases page, Podman Desktop, or another package manager — counts as external. Both can be present at once, because Scoop puts its shims on the **user** PATH while an MSI writes to the **machine** PATH. Which `podman.exe` runs then depends on the console window: an elevated console, a console opened before the installer ran, and a console under a different account can each resolve a different binary, and therefore a different version.

An external Podman still works, but the installer treats it as hands-off:

- The installer does not update it, because it does not own the installation.
- The last step of destructive recovery is skipped, so a broken Podman must be repaired by hand.
- Both installations share the connection configuration under `%APPDATA%\containers\`, so a machine created by one version can be unreachable from the other.

Resolve:

1. List every `podman` the PATH resolves:

```
Get-Command podman -All | Select-Object Source, Version
```

2. If more than one appears, keep the one under `scoop\shims` and remove the others through **Settings > Apps > Installed apps**. Uninstall Podman Desktop as well if it is present: it brings its own Podman.
3. Close every open console window. A window opened before the change keeps the old PATH and keeps resolving the removed binary.
4. Open a new console and confirm that only one entry remains.
5. Re-run the **Prepare Installer** step.

> ⚠ Run step 1 in the same kind of console the installer runs in. An elevated console and a normal one produce different answers, and that difference is itself the symptom.

4.8.8 Podman: Installation Is Incomplete

Symptom: preparation fails with *Podman environment verification failed. Your Podman installation may be incomplete.*

Cause: the installer checks two things before it uses Podman. First, that `win-sshproxy.exe` is present in the Podman installation folder — it forwards the machine's API onto a Windows named pipe, so nothing reaches Podman without it. Second, that `containers.conf` points at the folder holding it. The installer writes that pointer itself, into `%APPDATA%\containers\containers.conf`:

```
[engine]
helper_binaries_dir = ["<podman-install-folder>"]
```

An interrupted install or update, or a hand-edited `containers.conf`, breaks one of the two.

Resolve:

1. Confirm the helper binary exists. For the Scoop installation the installer manages, that is:

   ```
   Get-ChildItem "$env:UserProfile\scoop\apps\Podman\current" -Recurse
   -Filter win-sshproxy.exe
   ```

2. Inspect the configuration file and check that `helper_binaries_dir` points at the folder from step 1:

   ```
   Get-Content "$env:APPDATA\containers\containers.conf"
   ```

3. If only `containers.conf` is wrong, re-run the **Prepare Installer** step. The installer rewrites the file itself, which is non-destructive and needs no setting change.
4. If `win-sshproxy.exe` is missing, the Podman installation itself is damaged. Reinstall Podman: turn on **Reinitialize Podman** and re-run preparation, or reinstall by hand when Podman came from somewhere other than the installer. See Podman: Recovery Modes.

4.8.9 Podman: Networking or DNS Fails Inside the Machine

Symptom: Podman runs, but pulls fail, the registry is unreachable, or name resolution fails from inside the machine while the host itself is fine. Typically appears after a WSL update, or when a VPN client is connected.

Resolve:

1. Test resolution from inside the machine:

   ```
   podman machine ssh -- "getent hosts registry.vizrt.com"
   ```

2. Reset the WSL network stack, which is what most often fixes it:

   ```
   wsl --shutdown
   ```

Then start the Podman machine again.

3. **If a VPN is connected**, disconnect it and retry. Split-tunnel and full-tunnel VPN clients frequently take over WSL's DNS. If the install must run over the VPN, reconnect it *after* the Podman machine is running, or set a DNS server explicitly in `%UserProfile%\.wslconfig`:

```
[network]
generateResolvConf = false
```

and provide `/etc/resolv.conf` inside the machine.

4. If the installation has to run with the VPN connected throughout, turn on **VPN Compatible Mode** in **Settings** (`VPNInstallation` in `appsettings.json`). The installer then creates the Podman machine with user-mode networking, which keeps the machine reachable when a VPN client takes over the host's routes.

> ⚠ User-mode networking is applied when the machine is **created**, so turning the setting on changes nothing about a machine that already exists. Turn it on first, then remove the machine and re-run the installer so that it recreates the machine with the setting applied:
>
> ```
> podman machine stop
> podman machine rm
> ```

5. If your registry uses an internal certificate authority, the CA must be trusted **inside** the Podman machine as well as on the host. See Registry Access.

4.8.10 Podman: Not Enough Memory or Disk

Symptom: the Podman machine starts, but the cluster fails to come up, or containers are killed.

On WSL2 the machine's resources come from WSL, **not** from `podman machine set` — that command reports success and changes nothing. Set them in `%UserProfile%\.wslconfig`:

```
[wsl2]
memory = 16GB
processors = 8
```

Then apply the change and start the machine again:

```
wsl --shutdown
podman machine start
```

Size the values against the cluster you configured, and leave the host enough memory for Windows itself. See Prerequisites → Installer Host.

4.8.11 Prerequisite Check Fails: "Cannot Reach Vizrt Registry"

Resolve: this is a connectivity problem from the installer host to the package registry.

1. Open a browser on the installer host and visit `https://registry.vizrt.com/` (or your internal mirror).
2. If your environment uses a proxy, configure WinHTTP proxy settings: `netsh winhttp set proxy <proxy>:<port>`.
3. If you are air-gapped, switch the workflow to **Offline installation** and use a bundle prepared elsewhere.

4.8.12 Cluster Install Fails: "Node Did Not Become Ready"

Symptom: cluster installation completes provisioning, but one or more nodes never report Ready.

Resolve:

1. Expand the failing step in the wizard to see the live log.
2. Common causes:
   - **IP conflict**: the static IP you allocated is already in use. Pick a different IP and retry.
   - **DNS unreachable**: Talos cannot resolve the package registry. Confirm DNS works from the node's network.
   - **Insufficient resources**: the Hyper-V host or vSphere cluster lacks RAM/CPU for the configured node sizes.
3. After fixing the underlying issue, click **Retry**. The installer picks up where it left off; you should not need to restart the wizard.

4.8.13 Hyper-V: Virtual Switch or VM Creation Fails

Symptom: on a Hyper-V install, provisioning fails while creating the virtual switch or the node VMs.

Resolve — virtual switch:

- The **public** switch needs a **physical network adapter facing your network**. If that adapter is already bound to another virtual switch, it cannot be reused — free it, or pre-create a switch with the expected name.
- Conflicts with leftover switches from a previous attempt can block creation. Remove stale VAM switches (for example, `VAMPublicSwitch`, `VAMPrivateSwitch`, a `NATSwitch`) in **Hyper-V Manager → Virtual Switch Manager**, then retry.

Resolve — VM creation:

- **Base path already in use**: a folder with the VM name already exists at the deployment path. Choose a different base path or delete the leftover folder.
- **IP already in use**: one of the node IPs (control-plane, worker, or VIP) is taken by another machine or a VM outside the selected context. Free the IP or change it in the advanced settings.
- **Disk I/O / out of space**: the Hyper-V host disk is full. Point the **Hyper-V base path** at a disk with more free space, or free space on the current one.

4.8.14 Nodes: Static IP, DHCP, or DNS Problems

Symptom: nodes fail to come up, or DNS shows as invalid.

Resolve:

- If DHCP is unreliable in your environment, **de-select DHCP** in the wizard and provide a valid static **IP, subnet and gateway** per node.
- A **valid, reachable DNS server is always required**. If a node's configured DNS is wrong or unreachable it may show an *INVALID IP* in its network configuration. Confirm the DNS servers you entered are reachable from the node network.
- If the control-plane IP changed after a reboot (DHCP lease change), clients can no longer reach the API server. Give the control plane a **fixed IP** (DHCP reservation or static) to avoid this.

4.8.15 Deployment Fails on a Single Package

Symptom: cluster install succeeded, but during package deployment one package fails while others succeed.

Resolve:

1. Expand the failing package to read the error.
2. Click **Skip and continue only** if the failing package is optional. Required-package failures must be fixed.
3. After fixing, return to the deployment screen and click **Retry** — only the failed package is re-attempted, not the whole installation.

4.8.16 Certificate (AD CS / Enterprise CA) Deployment Fails

Symptom: deployment fails at the certificate/issuer step, with an error that the issuer resource is invalid — for example that `spec.url`, `spec.templateName`, or `spec.caBundle` is the wrong type, or "Error deploying Cluster Issuer".

Cause: this almost always means one of the **AD CS** inputs was left blank or mis-typed, so a placeholder was not substituted with a real value.

Resolve:

1. Re-open the **TLS certificate** step and confirm every AD CS field is filled: the **ADCS server URL**, the **CA bundle**, the **certificate template** name and the enrollment **username / password**.
2. Confirm the account can request certificates from that template on your AD CS server, and that the URL is reachable from the cluster.
3. Re-run the deployment. See Existing / native Kubernetes → Certificates for the full field list.

4.8.17 Sanity Check Shows "Endpoint Not Reachable"

Symptom: deployment succeeded, but the post-install sanity check reports your public hostname is unreachable.

Resolve:

- Confirm the DNS A record for your hostname points to a control- plane node IP.
- Confirm port 443 is open from your test location to that IP.
- If using **Auto-generate** TLS, your browser warns about the self-signed certificate — that's expected, but `curl` and similar tools may refuse. Test with a browser to rule TLS out.

4.8.18 Sign-in and Authorization

These apply when you configured an **external OIDC provider** (Microsoft Entra ID, Okta, Auth0, external Keycloak). The full setup is in the Identity provider setup guide; the table below maps the symptoms you are most likely to hit back to the setup step that fixes them.

Symptom	Likely cause and fix
Redirected to the provider, sign-in succeeds, then an error page about a **redirect URI / reply URL mismatch**.	The callback URL is not registered. Add the exact URI (scheme, host, port and path) to the app's redirect URIs — see step 1.
Sign-in succeeds but **every VAM action is denied** / the UI loads empty.	The user has no `vam` role. Assign the role (Entra: step 4 + step 7).
Upload/download fails with **access denied from storage** although the user is signed in.	The user lacks a storage role, or the role value does not match a storage policy name. Use exactly `readwrite` / `consoleAdmin` — see Object storage trust.
Users are **prompted to consent** individually, or app permissions are blocked.	Admin consent was not granted. Grant it in API permissions.
Sign-in worked for weeks, then **all sign-ins fail** at once.	The client secret expired. Create a new secret (step 5) and update it in **Platform → Identity**.
Token validation fails / role claim is missing (Entra).	Access tokens are still v1. Set `requestedAccessTokenVersion` to `2` — see step 6.
Background jobs fail to access storage even though interactive use works (Entra).	The On-Behalf-Of flow could not re-mint a token. Confirm the application exposes an Application ID URI and that admin consent has been granted (steps 2–3).

Verify the live settings. *After install you can review the resolved authority, token endpoint, scope and role claim in* ***Platform → Identity*** *in the Configuration Service. Compare them against the values table.*

4.8.19 Registry Access

For an offline install, the wizard can fail to load packages even though you selected a bundle.

Resolve:

- Make sure the bundle was produced by the **same major version** of the installer you are running.
- The bundle path must be accessible to the installer host (a UNC path is fine, as long as the installer process has read access).

- If the bundle is on removable media, copy it to a local disk and point the installer at the local copy.

4.8.20 Inspecting the Cluster Directly

If you have `kubectl` access to the cluster (the installer writes a `kubeconfig`, or use your own for a bring-your-own cluster), these read-only checks quickly locate a failing component:

```
kubectl get nodes -o wide                # node status, roles, IPs, versions
kubectl get pods -A -o wide              # pod health across all namespaces
kubectl get deployments -A               # READY vs DESIRED replicas
kubectl describe pod -n <ns> <pod>       # events, restarts, image-pull errors
kubectl logs -n <ns> <pod> --tail=100    # recent logs (add --previous for a crash loop)
kubectl get events -A --sort-by=.metadata.creationTimestamp   # recent cluster events
kubectl get svc,httproutes -A            # ingress / gateway routing
kubectl get secrets -A | Select-String domain-tls             # the gateway TLS cert secret
```

On Talos-based clusters `kubectl top` needs the metrics server and may report Metrics API not available — use `talosctl dashboard` for live node CPU/memory instead. Prefer these read-only commands; avoid editing cluster resources by hand except on Vizrt support's guidance.

4.8.21 Exporting Logs for Support

From any screen, use **Help → Export logs**. The installer creates a zip at the path it shows you, containing:

- Console output for the entire session.
- All wizard step logs.
- Cluster diagnostic output (`kubectl` describe, recent events).
- A redacted copy of your wizard answers.

Attach the zip to your support case. Logs do not contain your WIBU license file or your OIDC client secret.

4.9 Identity Provider Setup

VAM signs users in with **OpenID Connect (OIDC)**. You choose the provider during installation (the **Identity provider** step of the wizard, see Running the installer). Two broad options exist:

- **Bundled Keycloak**: the installer deploys and configures a Keycloak instance for you. It is **not an officially supported deployment option**, but it is fully pre-configured and is expected to work out of the box, which makes it the easiest way to get going. Nothing in this guide is required; skip straight to Signing in.
- **External OIDC**: you point VAM at an identity provider you already run: **Microsoft Entra ID** (Azure AD), **Okta**, **Auth0**, or your own **Keycloak**. This requires some one-time setup *in your provider* before you run the installer.

> ***Two different "Microsoft" integrations — don't confuse them.*** *This page is about **sign-in** (Entra ID / OIDC). If instead you are looking for how VAM obtains its **TLS certificate** from **Microsoft AD Certificate Services (AD CS)**, that is the certificate step, not identity — see Existing / native Kubernetes → Certificates and the TLS certificate options.*

This guide walks through that one-time setup. Most of it is written for **Microsoft Entra ID**, because Entra needs the most explicit configuration; the other providers section at the end covers the differences for Okta, Auth0 and external Keycloak.

> ***Do this before you run the installer.*** *The installer's Identity provider step asks for the IDs and secret you create here. Having them ready makes the install a single pass.*

4.9.1 How VAM Uses Your Identity Provider

VAM is more than a single web app, so understanding what it asks of your provider helps the setup make sense:

VAM component	What it does with the token
Configuration UI	Interactive browser sign-in (authorization-code + PKCE). The user's token is the entry point.
VAM API / services	Validate the inbound user token, then act **as that user** for every downstream call.
Object storage (MinIO/S3)	Exchanges the user token for short-lived, user-scoped storage credentials (STS `AssumeRoleWithWebIdentity`).
Message broker (RabbitMQ)	Validates the same token for management/messaging access.
Background work	Re-mints a correctly-audienced token *on behalf of* the original user (Entra **On-Behalf-Of**).

Two consequences fall out of this and drive the whole setup:

1. **Roles travel in the token.** VAM authorizes users from a role claim. The same role values are also what object storage maps to storage policies. Get the roles right and everything downstream works; get them wrong and the user signs in but can do nothing.

2. **VAM must be able to act on behalf of the user.** On Entra, access tokens are *audience-locked* — a token minted for the UI is not automatically valid for the API or for storage. VAM therefore needs to be a **confidential client with a certificate credential** so it can perform the On-Behalf-Of exchange. The installer provisions that certificate; you only need to allow the flow.

4.9.2 Microsoft Entra ID

Create **one app registration** that backs every VAM client (the UI, the API, the broker and the storage callback all use the same application). The steps are:

1. Register the application
2. Expose an API and add scopes
3. Add API permissions and grant consent
4. Define app roles
5. Add a client secret
6. Force v2 access tokens
7. Assign users and groups to roles
8. Collect the values for the installer

You need an Entra account with permission to create app registrations and grant admin consent (an **Application Administrator** or **Cloud Application Administrator**, or a Global Administrator).

Throughout, replace `<vam-host>` with the public hostname you will give VAM (for example `vam.example.com`).

1. Register the Application

1. In the Entra admin center, go to **Identity → Applications → App registrations → New registration**.
2. **Name**: `VAM` (or any name you recognize).
3. **Supported account types**: *Accounts in this organizational directory only* (single tenant) unless you have a specific multi-tenant requirement.
4. **Redirect URI**: select **Single-page application (SPA)** and enter the Configuration UI callback:

```
1  https://<vam-host>/config/auth/callback
```

5. Click **Register**.

After registration, open **Authentication** and add the remaining redirect URIs VAM uses. Add each under the platform shown:

Platform	Redirect URI	Used by
Single-page application	`https://<vam-host>/config/auth/callback`	Configuration UI
Single-page application	`https://<vam-host>/auth/callback`	VAM web apps
Single-page application	`https://<vam-host>/pilot/*`	Viz Pilot Edge SPA

Platform	Redirect URI	Used by
Single-page application	`https://<vam-host>/template-builder/*`	Template Builder SPA
Single-page application	`https://<vam-host>/data-server-config/*`	Data Server Config SPA
Web	`https://<vam-host>:30442/minio/ui/oauth_callback`	Object-storage console
Web	`https://<vam-host>:30443/oauth_callback`	Message-broker console

The exact ports for the storage and broker consoles depend on your ingress configuration. The values above are the defaults; if you changed them, adjust accordingly. You can also add these later — they are only needed for the respective admin consoles, not for normal VAM use.
*The three **Pilot Edge** SPA redirect URIs are only required if you deploy Viz Pilot Edge / Template Builder / Data Server Config. Entra does not accept `*` wildcards in SPA redirect URIs — register the concrete callback for each SPA (the SPA's served path, for example, > `https://<vam-host>/pilot/`). See Pilot Core Service and Pilot Edge.*

From the **Overview** page record the **Application (client) ID** and the **Directory (tenant) ID** — you need both for the installer.

2. Expose an API

VAM validates tokens against its own Application ID URI, so the application has to expose one.

1. Open **Expose an API**.
2. Next to **Application ID URI**, click **Add** and accept the default `api://<client-id>`.
3. Add a scope named `management-ui` if you want browser sign-in to the message-broker management console. No other scope is required.

The installer requests the `<client-id>/.default` scope, which aggregates all scopes and app permissions you have granted. You do not have to wire each scope into the installer individually — just make sure they exist and are consented (next step).

3. API Permissions

1. Open **API permissions**.
2. Confirm the **Microsoft Graph → Delegated** permissions include: `openid`, `profile`, `email`, `offline_access` and `User.Read`. Add any that are missing (**Add a permission → Microsoft Graph → Delegated permissions**).
3. Click **Grant admin consent for <your tenant>**. Every permission row must show a green *Granted* state.

If you skip admin consent, each user is prompted to consent individually — and for app permissions that require admin consent they are simply blocked. Always grant admin consent here.

4. Define App Roles

Roles are how VAM authorizes users. The **value** of each role is what appears in the token's `roles` claim, and VAM (and object storage) match on that value — so the values below must be exact.

1. Open **App roles → Create app role**.
2. Create at least the roles your deployment needs:

Display name	Value	Allowed member types	Purpose
VAM Service access	`vam`	Users/Groups	**Required.** Grants access to the VAM API. Without it, sign-in succeeds but every API call is rejected.
Storage read/write	`readwrite`	Users/Groups	Read/write access to object storage (maps to the storage `readwrite` policy).
Storage admin	`consoleAdmin`	Users/Groups	Full object-storage administration (maps to the storage `consoleAdmin` policy).
Configuration admin	`vam-admin`	Users/Groups	Access to the Configuration Service. The Configuration Service requires this exact role on every endpoint — there is no separate read-only tier.

Set every role's **state** to **Enabled**.
If you deploy **Viz Pilot Edge / Pilot Core Service**, add these roles as well (their values must be exact — Pilot Core Service authorizes on them directly):

Display name	Value	Allowed member types	Purpose
Pilot administrator	`pilot-admin`	Users/Groups	Full Pilot Edge administration (PCS Admin, Template Builder, Data Server settings).
Pilot editor	`pilot-editor`	Users/Groups	Create and edit Pilot data elements and templates.

Display name	Value	Allowed member types	Purpose
Pilot journalist	`pilot-journalist`	Users/ Groups	Edit Pilot data elements (journalist workflow).
Graphic designer	`graphic-designer`	Users/ Groups	Template Builder / graphics authoring.
Pilot MSE (read)	`pilot-mse`	Applications	Read-only access for the Media Sequencer Engine and other Vizrt services.

See Pilot Core Service and Pilot Edge for how these roles map to Pilot's authorization policies and for the collection-scoped role convention.

The `vam` role is mandatory. *It is the gate the VAM API checks on every request. The storage role values (`readwrite`, `consoleAdmin`) must match the names of the object-storage policies they map to (see Object storage trust). The Pilot role values (`pilot-admin`, `pilot-editor`, `pilot-journalist`, `graphic-designer`, `pilot-mse`) must match exactly — Pilot Core Service checks them by name.*

5. Client Secret

VAM uses a certificate for the On-Behalf-Of exchange (the installer provisions it), but a client secret is still required for the standard confidential-client configuration.

1. Open **Certificates & secrets → Client secrets → New client secret**.
2. Give it a description (`VAM`) and an expiry that matches your rotation policy.
3. Click **Add** and **copy the secret value immediately** — Entra shows it only once. This is the value you enter as the OIDC client secret in the installer.

*Set a calendar reminder before the secret expires. When it lapses, sign-in and token exchange break until you create a new secret and update VAM's **Platform → Identity** settings.*

6. Force v2 Access Tokens

VAM expects v2.0 access tokens (the `roles` claim and the `v2.0` issuer depend on it).

1. Open **Manifest**.
2. Find `requestedAccessTokenVersion` and set it to `2`:

```
1  "requestedAccessTokenVersion": 2
```

3. **Save**.

7. Assign Users

Because the app exposes app roles, Entra by default only lets **assigned** users sign in and only assigned users receive role claims.

1. From the app registration's **Overview**, click the linked **Managed application** (this opens the matching *Enterprise application*).
2. Open **Users and groups → Add user/group**.
3. Select the users or groups and assign the appropriate role (`vam` plus whatever storage/config roles they need).

*Assign roles to **groups** rather than individual users where you can — it is far easier to manage at scale. A user with no role assignment can authenticate but has no access.*

8. Values for the Installer

When you reach the installer's **Identity provider** step, choose **External OIDC**, select **Microsoft Entra ID** as the provider and enter:

Installer field	Value
Provider	`Entra`
Host	`https://login.microsoftonline.com`
Tenant / App ID	your **Directory (tenant) ID** (from step 1)
Client ID	your **Application (client) ID** (from step 1)
Client secret	the secret value from step 5

From these the installer derives the values VAM actually uses:

Derived value	Form
Authority	`https://login.microsoftonline.com/<tenant-id>/v2.0`
Token endpoint	`https://login.microsoftonline.com/<tenant-id>/oauth2/v2.0/token`
Discovery URL	`…/<tenant-id>/v2.0/.well-known/openid-configuration`
Requested scope	`openid profile email offline_access <client-id>/.default`

Derived value	Form
Role claim	`roles`

You do not type these in — they are listed so you can verify them in **Platform → Identity** after install, or in a support case.

4.9.3 Object Storage Trust

VAM's object storage (MinIO / S3) issues each user short-lived, user-scoped credentials by validating the user's token directly (STS `AssumeRoleWithWebIdentity`). For that to work, the storage layer must:

1. **Trust your Entra issuer.** The installer configures the storage OIDC settings to the same authority and client ID you entered above. No manual step is required for a standard install.
2. **Map the role claim to a storage policy.** Storage reads the `roles` claim and looks for a policy of the same name. This is why the app-role **values** in step 4 must match storage policy names:

Role value	Storage policy	Effect
`readwrite`	`readwrite`	Read and write objects.
`consoleAdmin`	`consoleAdmin`	Full storage administration.

If a user can sign in to VAM but uploads or downloads fail with an access denied error from storage, the cause is almost always a missing or mis-named storage role — check that the user carries a role value matching a storage policy. See Troubleshooting → Sign-in and authorization.

4.9.4 Pilot Core Service and Pilot Edge

If your deployment includes **Viz Pilot Edge** and the **Pilot Core Service (PCS)**, there is some extra identity-provider setup on top of the base VAM configuration above. Skip this section entirely if you do not deploy Pilot.

Pilot has three moving parts that touch the identity provider:

Part	Sign-in style
Pilot Edge browser SPAs	Three single-page apps — **Pilot**, **Template Builder**, **Data Server Config** — sign in with authorization-code + PKCE.
Pilot Core Service (PCS)	Validates the user's access token on every API call: checks the **audience**, then authorizes on **Pilot roles**.
Media Sequencer / services	Other Vizrt services call PCS read-only with a service token carrying the `pilot-mse` role.

1. Register (or Reuse) Redirect URIs for the Three SPAs

The three Pilot SPAs are served under `/pilot/`, `/template-builder/` and `/data-server-config/`. Their callbacks are the redirect URIs you already added in step 1:

```
https://<vam-host>/pilot/
https://<vam-host>/template-builder/
https://<vam-host>/data-server-config/
```

*On Entra, register each as a **concrete** SPA redirect URI (the served path of the app). Entra rejects `*` wildcards for SPA platform URIs — the `/pilot/*` form shown in the bundled-Keycloak realm only works for Keycloak. PCS's own `sso.json` maps all three SPAs to a single client ID, so they can share the VAM app registration (or you can give Pilot its own registration; either way the redirect URIs above must be present).*

2. Define the Pilot Roles

PCS authorizes every request against the `roles` claim, by exact role-value match. Add the five Pilot roles from step 4 (`pilot-admin`, `pilot-editor`, `pilot-journalist`, `graphic-designer`, `pilot-mse`). They map to PCS's authorization policies like this:

PCS policy	Satisfied by role(s)	Used for
Pilot admin	`pilot-admin`	PCS Admin, Template Builder, Data Server settings.
Graphic designer	`graphic-designer`, `pilot-admin`	Template / graphics authoring.
Pilot journalist	`pilot-journalist`, `pilot-editor`, `graphic-designer`, `pilot-admin`	Create / edit Pilot data elements.
Pilot read-only	`pilot-mse`, `pilot-journalist`, `pilot-editor`, `graphic-designer`, `pilot-admin`	Read endpoints used by MSE and other Vizrt services.

*The roles are **additive** in capability: `pilot-admin` satisfies every policy, `pilot-mse` only the read-only one. Assign `pilot-mse` to the service principal (MSE / integrations), not to people.*

3. Make Sure PCS Receives a Usable Audience

PCS rejects a token unless it validates the **issuer** *and* the **audience** (`aud`). Two consequences:

- **Bundled Keycloak:** handled for you. The realm's `audience` client scope stamps the `viz-am` audience onto user tokens, and the installer sets PCS's expected audience to match.
- **Entra (or other external OIDC):** the access token the SPA obtains must carry an audience PCS is configured to accept. In practice this means requesting the Pilot/PCS API scope so Entra issues a token audienced for it. The installer records this as PCS's `SSO:Audience`; the SPAs request it through an **extra scope** of the form `<pcs-api>/.default` (for example `VizPilot/.default`).

*If PCS returns **401 Unauthorized** for a user who can sign in to the rest of VAM, the token almost always has the wrong audience — the SPA obtained a token audienced for the UI, not for PCS. Check that the Pilot/PCS scope is being requested and that PCS's `SSO:Audience` matches the `aud` in the issued token.*

4. Collection-Scoped Roles (Optional, Advanced)

Beyond the five flat roles, PCS supports **per-collection** access: any role whose value begins with the prefix `pilot-collection` (for example `pilot-collection-sports`) grants the user access to the matching Pilot collection. This is opt-in — define such roles only if you use Pilot collection-level segregation, and make the suffix match your collection names. The prefix itself is configurable (`SSO:PilotCollectionPrefix`, default `pilot-collection`).

4.9.5 Other Providers

Everything above is written for Microsoft Entra ID. VAM also supports three other provider families. Before you invest time configuring one, check how complete that support is:

Provider	Interactive sign-in	Object-storage (MinIO)	Background / delegated work (token re-mint)	Support status
Bundled Keycloak	✅	✅	✅ (RFC 8693, pre-configured)	**Not officially supported — but pre-configured and expected to work**
Microsoft Entra ID	✅	✅	✅ (On-Behalf-Of, certificate)	**Supported**
External Keycloak	✅	✅	✅ *if* token exchange is enabled in your realm	**Supported**
Okta	✅	✅ (group claim)	⚠️ Limited — see notes	**Preview**
Auth0	✅	⚠️ Needs a custom claim	❌ No standard token exchange	**Preview**

> ***What "Preview" means.*** *Interactive sign-in to the Configuration UI works for every provider in the list. The caveats are about the two things VAM does after sign-in: mapping the role/group claim to object-storage policies, and re-minting a token for background work (transcode, import, async indexing). For Okta and Auth0 those paths are not yet validated end-to-end — read the per-provider notes before committing to them for production.*

The runtime only distinguishes **Entra** from **everything else**: a non-Microsoft issuer is treated as a Keycloak-style provider, and the delegated token re-mint uses the Keycloak **RFC 8693 token-exchange** call. That is why external Keycloak is fully supported, while Okta and Auth0 (which do not implement RFC 8693 the same way) fall back to running the user's original token and degrade to the service identity for background jobs.

Local Keycloak for Development (Not Officially Supported)

If you are a developer running VAM locally (outside the installer), you can point it at a standalone Keycloak you run yourself — for example a `quay.io/keycloak/keycloak` container started in dev mode. This is a **development-only** convenience and is **not an officially supported deployment option**: for any real install, use the **bundled Keycloak** the installer provisions, which is configured correctly out of the box.

To stand one up for local development:

1. Run Keycloak in dev mode and create a realm named `vam`.
2. Import the realm shape VAM expects (see the bundled realm below) — the clients `viz-am`, `swagger`, `rabbitmq`, `vue-config-web-app` and `pilot-edge`, the `policy` claim mapper and the audience mappers. The simplest path is to export the bundled realm from a throwaway installer run and import it into your local Keycloak.
3. Point VAM's `OidcHost` at the realm URL (`http://localhost:8080/realms/vam`).

Because none of the installer's TLS, CA-trust or ingress wiring applies here, you are responsible for making the realm reachable from every VAM component. Treat broken background/STS flows in this setup as expected, not as bugs.

External Keycloak

Use this when you already run your own Keycloak and want VAM to use it instead of the bundled instance. VAM treats it exactly like the bundled Keycloak — same fixed client IDs, same claim — so your realm must **mirror the bundled realm's shape**. This is the most involved part: the installer does not let you rename these clients.

1. **Create a realm** (any name; you will give VAM its full realm URL).
2. **Create the clients VAM expects**, with these exact client IDs:

Client ID	Type	Used by
`viz-am`	Confidential (client secret)	VAM API / services (also OBO/exchange)
`vue-config-web-app`	Public (PKCE)	Configuration UI browser sign-in
`swagger`	Public (PKCE)	API explorer

Client ID	Type	Used by
`rabbitmq`	Confidential	Message broker validation
`pilot-edge`	Public (PKCE)	Pilot Edge / Template Builder / Data Server Config (only if you use Pilot)

3. **Redirect URIs**: add the same callback URLs listed in section 1 (config UI, MinIO console, Swagger and the Pilot paths if applicable) to the matching clients.
4. **Realm roles**: create the VAM roles and storage policies as realm roles: `vam` (mandatory API gate), your storage policy names (for example, `readwrite`, `consoleAdmin`) and the Pilot roles (`pilot-admin`, `pilot-editor`, `pilot-journalist`, `graphic-designer`) if you use Pilot. See section 4 for what each role does.
5. **Claim mapper**: add a *realm-roles* mapper that emits the roles into a claim named `policy` (token claim name `policy`, multivalued, added to the access token). VAM and MinIO both read `policy`.
6. **Audience mappers**: stamp the `viz-am` and `rabbitmq` audiences onto issued tokens so the API and broker accept them.
7. **Enable token exchange** on the `viz-am` client if you want background/delegated work to act as the original user (VAM uses RFC 8693). Without it, async jobs fall back to the service identity.
8. **Installer values**: set **Provider** = `Keycloak`, **Host** = the full realm URL (for example, `https://keycloak.example.com/realms/vam`) and the `viz-am` **client secret**. The other client IDs are fixed, so there is nothing else to enter.

Private CA. *The installer only mounts its CA trust bundle for the bundled Keycloak. If your external Keycloak presents a certificate from a private CA, VAM components must already trust that CA at the OS level, or TLS to the realm's discovery and token endpoints fails.*

Okta

Okta works as a standard-discovery provider: the host you enter is the issuer/authority verbatim, the token endpoint is `<host>/v1/token` and the **group** claim (`groups`) carries authorization. Unlike Keycloak, **one Okta app's client ID is reused for every VAM client and audience.**

1. **Create an OIDC application** of type *Web* (it needs a client secret for the confidential flows).
2. **Sign-in redirect URIs**: add the VAM callback URLs from section 1 (config UI, MinIO console, Swagger).
3. **Groups**: create groups whose **names exactly match** the VAM roles and storage policies you need: `vam` (mandatory), your storage-policy names (for example, `readwrite`, `consoleAdmin`) and any Pilot roles. Assign users to the groups.
4. **Groups claim**: in the app's *OpenID Connect ID Token* / access token settings (or your custom authorization server's claims), add a claim named `groups` that includes the groups and make sure it is emitted on the **access token** (not only the ID token), because VAM and MinIO authorize from the access token.
5. **Installer values**: **Provider** = `Okta`, **Host** = your Okta issuer (your org URL, or `https://<org>.okta.com/oauth2/<authServerId>` if you use a custom authorization server), plus the app's **client ID** and **client secret**. Enter the same client ID for the shared/SPA client ID field.

Preview caveats. *The access-token audience VAM validates is the app's own client ID, and the request scopes are the standard OIDC set (`openid profile email offline_access`) — there is no resource scope for the broker, so RabbitMQ OAuth may need tuning. Background/delegated re-mint uses RFC 8693 `requested_subject`, which Okta's token-exchange grant does not support, so async jobs run as the service identity.*

Auth0

Auth0 is also a standard-discovery provider: token endpoint `<host>/oauth/token`, discovery at `<host>/.well-known/openid-configuration`.

1. **Create a *Regular Web Application*** (confidential, has a client secret).
2. **Allowed Callback URLs**: add the VAM callback URLs from section 1.
3. **Roles / authorization**: model your VAM roles (`vam`, storage policies, Pilot roles) however you prefer (Auth0 Roles, app metadata, or a database), then add them to the token with an **Action**.
4. **Claim**: VAM reads the policy/role claim named `policy`. Auth0 strips non-namespaced custom claims from tokens, so an Action that simply adds `policy` may not survive. This is the main limitation (see below).
5. **Installer values**: **Provider** = `Auth0`, **Host** = your tenant domain (`https://<tenant>.auth0.com`), plus the application's **client ID** and **client secret**.

Preview caveats — read before choosing Auth0. *Two gaps currently make Auth0 unsuitable for production: (1) VAM expects the role claim to be named exactly `policy`, but Auth0 only reliably emits namespaced custom claims, so object-storage policy mapping may not resolve; and (2) Auth0 does not implement RFC 8693 token exchange, so VAM cannot re-mint a delegated token for background work — those jobs run as the service identity. Interactive sign-in works, but treat Auth0 as a preview integration only.*

4.9.6 Changing the Provider After Install

You can change identity-provider settings later in the Configuration Service under **Platform → Identity** — for example to rotate the client secret or point at a different tenant. Sign-in is briefly interrupted while the change rolls out. If you switch provider family (say Keycloak → Entra), re-create the role assignments in the new provider first so users are not locked out.

4.9.7 Related

- Prerequisites → Identity provider
- Running the installer
- Configuration Service → Signing in
- Troubleshooting → Sign-in and authorization

4.10 VMware vSphere Deployment

This guide covers installing VAM on **VMware vSphere**, where the installer provisions the Talos Kubernetes nodes as virtual machines on a vCenter-managed cluster. It expands on the vSphere items in Prerequisites and is used alongside the main Running the installer walkthrough — the only difference is the **Kubernetes → Select** step, where you choose *New cluster on VMware vSphere*.

> ***Who this is for.*** *A vSphere administrator preparing the environment and permissions, and the operator running the VAM installer. If the installer is provisioning nodes on Hyper-V or you are bringing your own Kubernetes cluster, this page does not apply.*

4.10.1 How It Works

The installer deploys a Talos Kubernetes cluster on vSphere:

1. Downloads the Talos OVA (from the Image Factory, or uses a cached copy) and imports it into a vCenter **content library**.
2. Deploys the control-plane and worker VMs from that template into a resource pool, on a **DHCP-enabled** network, in Talos maintenance mode.
3. Detects each VM's IP address via VMware Tools.
4. Applies the Talos configuration to each node, bootstraps the cluster and retrieves the cluster credentials.

Re-runs are safe: if the content library, OVA template, resource pool, or VMs already exist from a previous attempt, the installer detects and reuses them rather than recreating them.

4.10.2 Environment Prerequisites

Set these up in vCenter **before** running the installer:

Requirement	Details
vCenter or ESXi host	Reachable from the installer host (for example, `https://vcenter.example.com`).
ESXi version	ESXi 6.7 Update 2 or later (VM hardware version `vmx-15`).
Datacenter	A vSphere datacenter that hosts the VMs.
Host / cluster	A specific ESXi host or cluster within the datacenter.
Datastore	A datastore with room for the OVA template plus all VM disks.
Network / port group	A network with **DHCP enabled** — VMs obtain their first IP via DHCP.
Content library	vCenter content libraries must be available (vCenter 6.5+).

Requirement	Details
Internet access	The installer host needs internet to download the Talos OVA (unless you did a prepare-offline bundle).
Credentials	A vSphere account with the required permissions.

Network Requirements

- The target VM network **must have DHCP** — the installer relies on it for initial IP assignment.
- The VMs must be reachable from the installer host on their DHCP addresses.
- VMware Tools must be able to report guest IPs back to vCenter (this is built into the Talos OVA).
- The installer host must be able to reach the VMs on port **50000** (Talos maintenance mode) and, after bootstrap, port **6443** (Kubernetes API).

Sizing

The installer applies the node sizing you choose in the wizard; the defaults are a good starting point:

Node type	vCPU	RAM	Disk
Control plane	2	4 GB	30 GB
Worker	4	8 GB	40 GB

A typical production layout (3 control plane + 4 workers) needs roughly 22 vCPUs, 44 GB RAM and 250 GB of disk in total, plus ~1 GB for the OVA template in the content library. Review sizing against your media volume with your Vizrt representative before committing hardware.

4.10.3 Required vSphere Permissions

The vSphere account used by the installer needs a role with the privileges below. For simplicity you can assign the role at the **datacenter** level with propagation to children; for tighter security, scope it to the specific datacenter, host/cluster, datastore, network, content library and resource pool the installer touches.

Recommended Custom Role: "VAM Installer"

Content Library

- Create local library
- Delete local library
- Update library item
- Add library item
- Deploy from OVF template
- Read storage

Virtual Machine — Inventory

- Create new
- Delete

Virtual Machine — Configuration

- Advanced configuration
- Change CPU count
- Memory
- Disk change

Virtual Machine — Interaction

- Power On
- Power Off

Virtual Machine — Guest Operations

- Guest operation queries (read guest IPs via VMware Tools)

Resource

- Assign virtual machine to resource pool
- Create resource pool

Datastore

- Allocate space
- Browse datastore

Network

- Assign network

Read-Only / Inventory

- `System.Read` (verify connectivity)
- `Inventory.Browse` (enumerate datacenters, hosts, pools, datastores, networks)

Where to Apply the Role

Apply with **propagation to children** on: the target **datacenter**, the **host / cluster** that runs the VMs, the **datastore** used for VM storage and the content-library backing, the VM **network / port group**, the **content library** and the target **resource pool** (or its parent host or cluster if the installer creates the pool).

4.10.4 Running the Installation

1. Prepare the vCenter environment and permissions above.
2. Launch `vamctl.exe` and follow Running the installer through the prerequisite and source steps.
3. At **Kubernetes → Select**, choose **New cluster on VMware vSphere**.
4. Enter the vSphere connection details. The wizard then lets you pick the **datacenter, host, datastore, network and resource pool** from live drop-downs populated from your vCenter.
5. Set the cluster topology and per-node sizing, review and start the install. The installer imports the OVA, deploys the VMs, waits for them to report IPs (up to ~10 minutes), applies the Talos configs and bootstraps the cluster.
6. Continue with the deployment and sanity-check steps as in the main walkthrough.

> All vSphere settings can also be supplied through `appsettings.json` or command-line arguments, so the installer can run unattended for automated deployments. Contact your Vizrt representative for the unattended-deployment reference.

4.10.5 Known Issues

VMXNET3 Network Adapter and Cluster Networking

With the default **VMXNET3** adapter, the cluster's pod network (Flannel vxlan) can fail due to hardware checksum-offload issues. Either:

1. **Use an Intel E1000 adapter** for the VMs, or
2. disable hardware offload on the pod-network interface with a Talos `EthernetConfig` patch:

```
apiVersion: v1alpha1
kind: EthernetConfig
name: flannel.1
features:
  tx-checksum-ip-generic: false
```

If you hit this, contact Vizrt support — the patch can be applied during installation.

OVA Download Failures (TLS)

The Talos Image Factory CDN uses post-quantum TLS that native Windows tooling may not support. The installer works around this automatically (falling back to a bundled `wget`). If OVA downloads fail, ensure the installer host has internet access, or use a **prepare-offline** bundle so the OVA is fetched ahead of time — see Offline installation.

4.10.6 Next Steps

- Post-install validation
- Configuration Service → Signing in

4.11 Offline (Air-Gapped) Installation

VAM can be installed on hosts that have **no internet access**. Because the air-gapped target cannot pull container images or packages from the Vizrt registry, you first produce a self-contained **offline bundle** on a machine that *does* have internet, transfer it to the target and run the installer against the bundle.

> ***Who this is for.*** *Operators installing VAM into a secured or disconnected network. If your installer host has internet access, use the standard online install instead — it is simpler.*

4.11.1 The Three Phases

```
Connected host (internet)          Removable media / file transfer
Air-gapped target
──────────────────────────────────────
──────────────────────────────────────────────────────
──────────────────────────
1. vamctl → Prepare offline  ──►  2. copy the offline bundle      ──►
  3. vamctl → Offline install
   (downloads everything)            (USB / DVD / secure copy)
(installs from the bundle)
```

1. **Prepare offline**: on a Windows host *with* internet, run `vamctl` and choose **Prepare offline installation**. It downloads everything the target needs and writes it into a single bundle.
2. **Transfer**: copy the bundle to the air-gapped target (USB drive, DVD, or an approved file-transfer path).
3. **Offline install**: on the target, run `vamctl` and choose **Offline installation**, pointing it at the transferred bundle.

The connected host and the target can be different machines; nothing from the build host is needed on the target beyond the bundle.

4.11.2 What the Bundle Contains

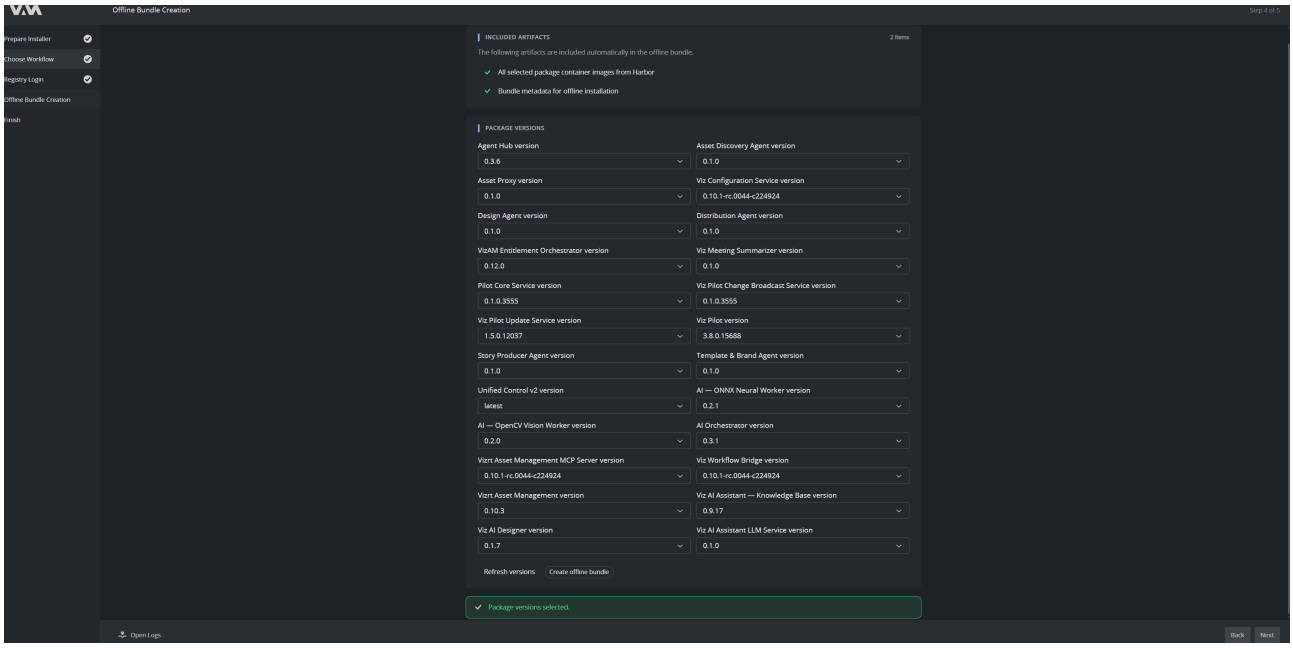
The prepare-offline step gathers three classes of asset so the target never has to contact a registry:

Asset	What it is
Talos OS images	The Talos installer image, boot ISO and (for vSphere) the OVA used to bring up the cluster nodes.
Container images	Every VAM service image plus the third-party images the platform depends on, packaged into a single image cache.
Packages	The VAM Helm / application packages (the catalog the installer deploys).

Everything sits next to the installer executable in the bundle, so the whole thing can be copied as one folder or archive.

4.11.3 Phase 1 — Prepare the Bundle (Connected Host)

1. On a Windows host with internet access, launch `vamctl.exe`.
2. Choose **Prepare Offline Bundle**.
3. Sign in to the registry, as for a normal install. See Running the Installer.
4. On **Offline Bundle Creation**, review the bundle contents. **Included artifacts** lists what is added automatically: the container images for every selected package, and the bundle metadata the target host needs.
5. Under **Package versions**, choose the version of each package to include. Every package the platform can install has its own version list, pre-filled with the current version from the registry. Use **Refresh versions** to re-read the registry. The panel confirms *Package versions selected* when the selection is valid.
6. Click **Create offline bundle**. The installer downloads the Talos images, builds the container-image cache and stages the packages. This takes a while and needs enough free disk for the whole bundle — budget well over the sizes in Prerequisites → Installer host.
7. When it finishes, you have a complete offline bundle folder.



The target installs exactly the versions you bundle here. To install newer versions later, produce a new bundle.

4.11.4 Phase 2 — Transfer to the Target

Copy the entire bundle to the air-gapped host using whatever transfer mechanism your environment allows (USB drive, DVD, or a controlled copy). Keep the folder structure intact — do not copy only parts of it.

4.11.5 Phase 3 — Install on the Target (Air-Gapped Host)

1. On the air-gapped host, launch `vamctl.exe` (from the bundle, or an installed copy).
2. Choose **Offline installation**.
3. When asked for the package source, point the installer at the **transferred bundle** (folder or archive) rather than the online registry. The installer switches to *local-only* mode — it never tries to reach `registry.vizrt.cloud`.
4. Continue exactly as in the standard walkthrough: prerequisite checks, cluster selection and configuration, deployment and the final sanity check. The cluster nodes are pre-seeded from the bundled image cache, so pods start without any external pulls.

If the installer reports that no image cache was found, the bundle is incomplete or the wrong folder was selected — re-check the transfer and point it at the bundle root.

4.11.6 Notes and Limitations

- **Licensing still applies.** An offline install still needs a valid WIBU license — a soft-license file or a reachable CodeMeter dongle. See Prerequisites → Licensing.
- **Identity provider.** If you use an external OIDC provider, the cluster still needs network access to *that* provider for sign-in, even when the install itself is offline. The bundled Keycloak avoids that external dependency.
- **VMware offline** provisioning has additional constraints; if you are targeting vSphere in an air-gapped environment, confirm the supported path with your Vizrt representative before you start.

4.11.7 Next Steps

- Post-install validation
- Configuration Service → Signing in

4.12 Existing Kubernetes Cluster

Instead of letting the installer create and manage a cluster for you (on Hyper-V or vSphere with Talos Linux), you can deploy VAM into a **Kubernetes cluster you already run** — on-prem or in the cloud. The installer then skips all VM / OS provisioning and only deploys VAM's platform components and services as Helm packages into your cluster.

This page covers the prerequisites for that cluster and how the install proceeds. For the summary requirements table see Prerequisites → Bring your own Kubernetes cluster.

> ***Who this is for.*** *A Kubernetes / platform administrator who owns an existing cluster and wants VAM deployed onto it. If you want the installer to build the cluster for you, use the Hyper-V or VMware / vSphere paths instead.*

4.12.1 What the Installer Deploys into Your Cluster

VAM is not a single container — it is a platform. When you point the installer at your cluster, it deploys (as packages) the components VAM depends on, alongside the VAM services themselves:

Layer	Component(s)	Role
Ingress / gateway	**Traefik** (Gateway API)	Terminates HTTPS and routes to the internal services.
Certificates	**cert-manager** + **trust-manager** + an issuer (self-signed / Let's Encrypt / **AD CS**)	Issues and renews the gateway certificate; distributes the CA bundle.
Object storage	**MinIO / AIStor** (operator + object store)	The authoritative media store (S3 API).
Block storage	**DirectPV** and/or **local-path-provisioner**	Local NVMe/SSD provisioning for the stateful workloads.
Databases / index	**SurrealDB**, **OpenSearch**	Cache / index over the object store.
Messaging	**RabbitMQ**	Event bus between services.
Identity	**Keycloak** (bundled) or your **external OIDC**	Sign-in for users and services.
Container registry	**Harbor**	In-cluster OCI registry / pull-through cache (where an internal one is used).
Observability	**kube-prometheus-stack** (Prometheus, Grafana, Alertmanager), **Loki** + **Promtail**, **OpenTelemetry Collector**	Metrics, logs, traces and dashboards.

Layer	Component(s)	Role
GPU (optional)	**NVIDIA device plugin**, **AMD GPU operator**	Exposes host GPUs to workloads that need them (for example, AI features).
VAM services	**VAM core**, **Configuration Service**, **Deployment Console**, **MCP server**, **Workflow Bridge**	The application itself.

You can let the installer deploy all of these, or — if your cluster already provides some of them (for example your own ingress controller or an existing `StorageClass`) — configure VAM to use what you have. Discuss the split with your Vizrt representative for a production design.

> ***Component versions are pinned per VAM release.*** *Each package declares the exact upstream chart / app versions it ships, so an installation is reproducible. As of the current release these include Traefik* `v3.7.6`*, cert-manager* `v1.21.1`*, trust-manager* `v0.24.0`*, the AD CS issuer* `2.1.4`*, Keycloak* `26.3.3`*, RabbitMQ* `4.3.4`*, OpenSearch* `3.7.0` *and SurrealDB* `3.2.0`*. The authoritative, always-current versions for* ***your*** *installation are shown per package in the* ***Configuration Service → Packages*** *view after install.*

4.12.2 Cluster Prerequisites

Area	Requirement
Kubernetes version	A currently-supported version — confirm the minimum with your Vizrt representative.
Access	An admin `kubeconfig`; the installer deploys into its **current context**. `kubectl` reachable.
Isolation	The cluster, or at least dedicated namespaces, not shared with unrelated workloads.
Capacity	Enough total CPU / RAM / disk for the worker sizing.
Storage	Block devices for **DirectPV**, or an existing default `StorageClass` (see below).
Ingress / exposure	A `LoadBalancer` service, or NodePort ingress, reachable by your clients.
DNS	An A record for the public hostname (for example, `vam.example.com`) pointing at the ingress.

Area	Requirement
TLS	A certificate strategy (self-signed, Let's Encrypt, provided cert, or **AD CS** — see below).
Identity	Bundled Keycloak, or an external OIDC provider configured per Identity provider setup.

Storage — DirectPV or a StorageClass

VAM's object store (MinIO / AIStor) needs fast local block storage.

- **DirectPV (recommended for local NVMe/SSD).** Installed via the `kubectl` **krew** plugin, DirectPV discovers and formats raw drives and exposes a `directpv-min-io` storage class. The high-level flow is `kubectl directpv install` → `kubectl directpv discover` → review the generated `drives.yaml` → `kubectl directpv init drives.yaml`. See the DirectPV documentation.
- **Existing `StorageClass`.** If your cluster already provides suitable persistent block storage, VAM can use it instead. Provision capacity sized to your media volume.

For storage spread across multiple nodes, use the MinIO Erasure Code Calculator to size the layout — your Vizrt representative can help.

Certificates — cert-manager Issuers

The gateway serves HTTPS using a certificate issued by `cert-manager`. Choose the issuer that fits your environment (this is the same choice as the installer's TLS certificate step):

- **Self-signed**: evaluation only.
- **Let's Encrypt**: if the hostname is publicly resolvable.
- **Provided certificate**: a PEM certificate + key you already hold.
- **Enterprise CA / AD Certificate Services (AD CS).** For on-prem Microsoft environments, VAM uses an **ADCS issuer** with `cert-manager`. You provide:
  - the **ADCS server URL**,
  - the **CA bundle** (root/intermediate chain),
  - the **certificate template** name to request against and
  - the **credentials** (username + password) of an account permitted to enroll certificates from that template.

  `trust-manager` then distributes the CA bundle so in-cluster clients (such as MinIO) trust the issued certificate.

Identity — Entra ID / OIDC

Sign-in works exactly as for a provisioned cluster. Use the bundled Keycloak for the simplest path, or configure an external provider — **Microsoft Entra ID**, Okta, Auth0, or your own Keycloak. The full provider setup (app

registration, app roles such as `vam` / `vam-admin` / `readwrite` / `consoleAdmin`, API permissions, redirect URIs, client secret and enforcing v2 tokens) is in Identity provider setup.

4.12.3 Running the Install

1. Set your `kubectl` context to the target cluster — the installer deploys into the **current context**.
2. Prepare storage, DNS, certificate inputs and (if external) the OIDC details ahead of time.
3. Launch `vamctl.exe` and follow Running the installer through the prerequisite and source steps.
4. At **Kubernetes → Select**, choose **Use an existing Kubernetes cluster** and point the installer at your `kubeconfig`.
5. The installer skips cluster provisioning and goes straight to **Deployment configuration** — set the public hostname, TLS option, identity provider and license, then deploy.
6. Finish with the sanity check as in the main walkthrough.

The same deploy-to-existing-cluster path can be scripted non-interactively with `appsettings.json` and command-line overrides (`DeploymentMode=Skip`). Contact your Vizrt representative for the unattended-deployment reference.

4.12.4 Next Steps

- Post-install validation
- Configuration Service → Signing in